Pazarlama Karması Modeli (Marketing Mix Model)

Faruk Ayaz[1]

[1]*Akdeniz Üniversitesi, Yönetim Bilişim Sistemleri*

Özet

Bu makalenin amacı, Pazarlama Karması Modelinin (Marketing Mix Model) ne olduğu, neler içerdiği, nasıl kurulacağı, ne gibi katkılar sağladığı ve gereksinimlerinin neler olduğu sorularının cevaplanması ve bu konu hakkındaki araştırma sonuçlarının paylaşılmasıdır. Makale kapmasında yaptığımız araştımalar ve deneysel işlemler için bir veri kümesi kullanmıştır. Elde edilen veri kümesi üzerinde keşifçi veri analizi, öznitelik seçimi ve parametre optimizasyonu işlemleri yapılmıştır. Veri analizleri sonrasında Python programlama dili kullanılarak makine öğrenmesi modelleri oluşturulmuş ve bu modeller arasından en iyi performansı elde edilen modellerle aşırı parametre optimizasyonu yapılmıştır. Aynı zamanda değişkenlerin esneklik değerleri hesaplanmıştır. Son olarak veri kümesi Optiscorer Engine'yle skorlanmış ve Pyhton üzerinde elde edilen değerle karşılaştırılmıştır.

Anahtar Kelimeler: *Pazarlama, Veri Bilimi, Veri Analizi*

Summary

The purpose of this article is to explain the Marketing Mix Model. It is the sharing of research results on what this modeling is, what it includes, how it will be set up, what contributions it provides and what its needs are. In our study, researches have been made on the Marketing Mix Model to address these questions. It used a dataset for the research and experimental procedures we have done in the article grab. Exploratory data analysis, feature selection and parameter optimization processes were performed on the obtained suitable data set. After data analysis, machine learning models were created using the Python programming language, and extreme parameter optimization was performed with the models that gave the best performance among these models. At the same time, the elasticity values of the variables were calculated. Finally, the data set was scored with Optiscorer Engine and compared with the value obtained on Pyhton.

Key Words: Marketing, Data Science, Data Analysis

1.Giriş ve Tanım

1.1 Pazarlama Karması Modeli Nedir?

Pazarlama Karması Modeli (Marketing Mix Model, MMM), çok değişkenli regresyonlar, satış ve pazarlama değerleri içeren istatistiksel analizlerdir[1]. Şirketlerin marka değerlerine göre tüketicinin tepkisi hakkında "ne yaparsam satışlarım artar" sorusunun yanıtlanmasını sağlar. İyi bir modelleme tutarlı veri kaydı, tüketici tepkisi, hem kontrol edilebilir (fiyat, reklam harcaması vb.) değişkenler hem de kontrol edilemeyen (mevsimsellik, gün, hafta vb.) değişkenler içermelidir [2]. Pazarlama karması modelinin kullanımı her geçen gün artmaktadır. Şirketler de modeli değerlendirmeye ve kullanmaya artan bir ilgi duymaktadırlar. Modelin kullanımının artması nedeniyle pazarlama karması modellemesi şirketlerin pazarlama bütçelerinde önemli bir rol almaya başlamıştır [3].

1.2 Veri Bilimi ve Pazarlama Karması Modeli

Pazarlama alanında yürütülen faaliyetlerin kayıt altına alınması veri biliminin pazarlama faaliyetlerinde kullanılmasına olanak sağlamıştır. Veri bilimi pazarlama faaliyetlerini belirlemede ve elde edilen sonuçların değerlendirilmesinde önemli rol oynamaya başlamıştır. Günümüzde Pepsi, Coca Cola, Starbucks gibi firmaların aktif olarak modellemeyi pazarlama faaliyetlerini belirleme de kullandıkları bilinmektedir [4].

Yapılan akademik çalışmalarla modelin işlevselliği konusunda araştırmalar yapılmıştır. Micheal J. Wolfe ve John C. Crotts [5] Çalışmalarında turizm sektöründe yapılan pazarlama yöntemlerinin etkisini ölçmek istemişlerdir. Makale kapsamında yapılan araştırmalarda kullanılan veri bir müzenin ziyaretçi veri kümesi kullanılmıştır. Müzeye yapılan daha önceki ziyaretlerinden oluşan veri kümesi kullanılarak bir model oluşturulmuştur. Bu modelden çıkan sonuçlara göre yeni bir pazarlama stratejisi belirlenip ve bu strateji uygulanmaya başlanmıştır. Yapılan bu pazarlama stratejisi 77 gün uygulanmış ve elde edilen yeni verilerin sonuçları bir önceki pazarlama stratejisine göre daha olumlu sonuçlar elde edildiği tespit edilmiştir .

Asgarnezhad Nouri ve Milad Soltani[6] makalelerinde kullanmış oldukları veri kümesi Kıbrıs'a tatile gelen turistlere sorulan sorular ve bu soruların cevaplarıyla oluşturulmuş bir veri kümesidir. Bu makaleye konu olan araştırma yabancı turistlerin Kıbrıs'ta tatil yapma konusundaki yönelimlerini olumlu yönde değiştirmeyi amaçlamaktadır. Makale sonucunda Kıbrıs'ta yabancı turistin artması için uluslararası seminerler ve uluslararası festivallere daha çok ev sahipliği yapması gerektiğini fikri ortaya konmuştur. Aynı zamanda turistlerin ülkeye gelebilmesi için vize ve rezervasyon işlemlerinin internet ortamında daha ulaşılabilir hale getirilmesi gerektiği sonucuna varılmıştır.

1.3 Makale Metadolojisi

Bu makalede yapılmak istenen araştırma veri bilimi tekniklerini kullanarak bir pazarlama karması modeli kurmaktır. Bu modeli kurmak için kullanacağım veri kümesi bir markanın belirli bir ürünün iki yıl boyunca haftalık toplam satışlarını ve o hafta içinde yapılan pazarlama harcamalarını içermektedir. Araştırmamızda kurmak istediğimiz modele uygun bir veri kümesi bulmak için Kaggle platformundan yararlanılmıştır. Aramalar sonucunda bulunan veri kümesi, keşifçi veri analizi faaliyetlerini gerçekleştirebilmek için Knime programı kullanılmıştır. Knime, kod yazmaya gerek duymadan veri analiz akışları çizebileceğimiz bir platformdur. Knime programında ön hazırlık işlemleri, öznitelik seçimi ve parametre optimizasyonu işlemleri yapılmıştır. Bu işlemler sonucunda elde edilen bilgiler doğrultusunda JupyterLab çalışma alanında Pyhton programlama dili kullanılarak uçtan uca veri analizi ve esneklik değerlerinin hesaplanma işlemleri yapılmıştır. Son olarak veri kümemizi AutoML yaklaşımıyla oluşturulan Optiscorer ürünüyle test edilip elde edilen bütün sonuçlar değerlendirilmiştir. Bu sonuçlar doğrultusunda pazarlama faaliyetlerinin satışlar üzerindeki etkilerini doğru olarak hesaplamak ve elde edilen sonuçların bir sonraki pazarlama stratejilerini düzenleme de kullanılabilir hale getirmek istenmektedir.

2. Pazarlama Karması Modelinin Hedefleri ve Gereksinimleri

Pazarlama Karması Modeli, projelerinin amacı daha önceki pazarlama performanslarını hesaplayarak gelecek pazarlama harcama getirilerini geliştirmektir. Modelden elde edilen sonuçlar pazarlama bütçesindeki taktikleri, ürünleri, segmentleri, zamanı ve pazarları yeniden düzenlememize yardım eder. Bütün pazarlama taktiklerinde kaliteli veri, yeterli zaman, ürün, demografik özellikler ve market çeşitliliği projeye dahil edilmelidir. Her bir proje veri toplamaktan başlayıp, optimizasyon ve gelecek stratejileriyle biten 4 farklı evreden oluşmaktadır. Bu evreler aşağıdaki gibidir.

1. Verinin Toplanması ve Bütünlüğü
2. Modelleme
3. Model Tabanlı İş Ölçümleri
4. Optimizasyon ve Simülasyon

2.1. Verinin Toplanması ve Bütünlüğü

Araştırmamız için istatiksel modellemeye uygun olacak şekilde çalışan verileri toplamamız gerekmektedir. Hangi ürünün analizinin yapılacağını belirlemek, hangi zaman aralığına bakılacağını belirlemek, kullanılacak birim zaman boyutunun belirlenmesi, hangi pazarın modellemesinin yapılacağını belirlenmesi, satış performansı ölçümlerinin belirlenmesi, marka marj oranları ve pazarlama taktik harcamaları maliyetleri belirlenmelidir.

2.2. Modelleme

Kullanılan istatistiksel model genel olarak satıcı tarafından belirlenir. Modelin işlevsel olması için reklam verenle beraber modelin sorularımıza doğru cevaplar verip veremeyeceğinden emin olmamız gerekmektedir. Marka müdürleri, Pazarlama Karması Modeli kurulurken kendi analiz ekipleriyle bu aşamada projeye dahil olmalıdır. Analiz ekibinin istatiksel detaylar ve özelleştirmelerin içinde bulunması çok önemlidir. Bu konuda oluşabilecek herhangi bir soru veya kaygı model kuran kişiyle paylaşılmalıdır.

2.3. Model Tabanlı İş Ölçümleri

Pazarlama Karması Modelinden elde ettiğimiz sonuçlarla bizim daha önceden cevaplanmasını istediğimiz sorularımızın karşılaştırılmasını yapmamız gerekmektedir. Projemiz satışımızı etkileyen her bir taktiğin sonuçlarını bize çıktı olarak verecektir ve bu sonuçları paylaşmadan önce satıcıyla görüşüp bu sonuçların daha önce cevaplamayı hedeflediğimiz sorulara cevap olup olmadığını kontrol etmemiz gerekmektedir.

Pazarlama Karması Modelinin temel çıktısı satışların ayrıştırılmasıdır. Her bir taktiğin satışlar üzerindeki etkisi gösterilir. Bu çıktı üzerinden farklılıkların asıl nedenlerini, artış sağlayabilecek pazarlama taktiklerini görebiliriz. Genel olarak bu nedenler satış takımı tarafından kontrol edilmeyen değerlerdir. Artış gözlemlenen taktikler ise tam tersidir pazarlama ve satış takımları tarafından kontrol edilebilir. Projemizde var olan bütün medya kampanya taktiklerinin artışlarını ölçümleyebilmemiz gerekmektedir. Bu şekilde var olan kampanyanın devam edip etmemesi gerektiğine karar verebileceğiz.

2.4. Optimizasyon ve Simülasyon

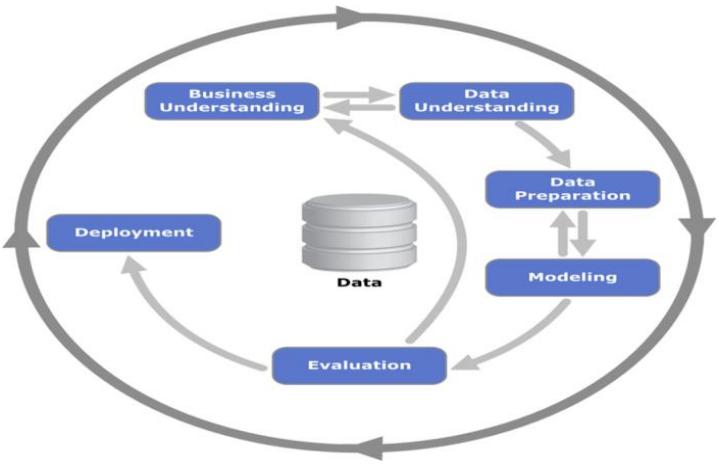
Pazarlama Karması Modellemesinin son evresi elde ettiğimiz yeni çıktıları bir sonraki pazarlama planlamasının yeni girdileri haline gelecektir. Modeli kurduktan sonra optimizasyon ve simülasyon denemeleri yapabilirisiniz. Denemeler bir sonraki model oluşturma konusunda fikirler sağlayacaktır. Bu denemeler her bir pazarlama taktiğinin satışa olabilecek etkisini gösterecektir. En iyi senaryoyu oluşturabilecek pazarlama planın girdilerini bize çıktı olarak verecektir. Bu işlemler sonucu elde ettiğimiz değerler bizim ihtiyaçlarımızı karşılayacak durumda olacaktır. [7]

3. Uygulama

Uygulamalı olarak veri analizi yaptığım kaynaklardan birisi Knime programıdır. Knime, kod yazmaya ihtiyaç duymadan programdaki nodları kullanarak uçtan uca veri analizi yapabilme imkanı sağlayan bir programdır. Uçtan uca veri analizini yapabilmek için takip edilmesi gereken bir süreç vardır. Bu sürece CRISP-DM (Cross Industry Standard Process Model for Data Mining) denir.

3.1 CRISP-DM

CRISP-DM süreci yaygınlaşan veri analizi kullanımdan sonra bu kullanımın daha etkili ve doğru bir şekilde yapılmasını amaçlayan bir süreçtir. Bu süreçte olan 6 aşamanın sırasıyla uygulanması halinde sizi doğru bir veri analiz sonucuna götürmesi mümkündür. CRISP-DM sürecinin kullanılmasının bir başka sebebi ise endüstride veri analizi konusunda bir standartlaşma sağlanmaya çalışılmasıdır.[8]



Grafik 1 - CRISP-DM

3.1.1 İşin Anlaşılması (Business Understanding)

İşin Anlaşılması, CRISP-DM sürecinin ilk aşamasıdır. Yapılacak olan projenin ne amaçla yapılmak istendiğini anlamak veri analizi aşamalarında çok önem taşımaktadır. Bu aşamada yapılacaklar işin anlaşılması, işe dair daha önce yapılan araştırmalara göz atılması ve iş sektöründe karşılaşabileceğimiz sorunların önceden saptanmaya çalışılması gerekmektedir.

3.1.2 Verinin Anlaşılması (Data Understanding)

Verinin Anlaşılması alanında istediğimiz değerlere ulaşabildiysek ve projenin amacını anlayabildiysek bir sonraki aşama olan Verinin Anlaşılması aşamasına geçebiliriz. Bu aşama ise amacımıza uygun verileri toplamamız, verilerin neyi temsil ettiğini anlamamız ve elde ettiğimiz verilerin amacımıza uygun olup olmadığına karar vermemiz gerekmektedir.

3.1.3 Veri Ön Hazırlık (Data Preprocessing)

Verilerimizi topladıktan sonra elde ettiğimiz verilerde modelimizi oluştururken sorunlar çıkartabilecek veri tipleri, uç değerler ve eksik verilere dikkat etmemiz gerekmektedir. Bu tür sorunlar modelimizi oluştururken model de hatalara ve yanlılığa sebep olabilir. Bu tür sorunlar çözülürken kurulan modelin amacı göz önünde bulundurulup ona göre çözümler bulunmaya çalışılmalıdır.

3.1.4 Modelleme (Modeling)

Bu aşamadan önce yapılan işlemlerden emin olunup bu aşamaya geçilmesi gerekmektedir. Veri Ön Hazırlık aşamasında yapılanlar yanlışlar modelleme işlemi sırasında sorunlar ortaya çıkartabilir. Modelleme işlemi yapılırken kurulan modelin veriye uygun olduğu gözlemlenmelidir. Model sonrası çıkacak bilgilerin anlamlı ve faydalı olmasına özen gösterilmelidir.

3.1.5 Değerlendirme (Evalutaion)

Bu aşamada bir önceki modelleme aşamasında ve kurulan modellerden elde ettiğimiz bilgiler doğrultusunda çıkan değerlerin bizim istediğimiz sonuçlar olup olmadığı ve hangi modelin bu isteklerimizi daha iyi yerinde getirdiği gözlenmeye çalışılır. Eğer çıkan sonuçlar bizim elde etmek istediğimiz cevapları vermiyorsa CRISP – DM aşamalarından ilki olan İşin Anlaşılması aşamasına geri dönüp ortadaki sorun anlaşılmaya çalışılır.

3.1.6 Uygulama (Deployment)

Bu aşama CRISP-DM sürecinin son aşamasıdır. Bir önceki aşamadan gelen bilgiler doğrultusunda oluşturmak istediğimiz son modeli oluşturabiliriz. Aynı zamanda proje ilerlerken ortaya çıkan yeni ihtiyaçlar ve istekler doğrultusunda yeni modelimize yön verebilir ve daha önceki sorularımıza cevaplar bulabiliriz.

3.2 Knime ile Keşifçi Veri Analizi

Makalenin amacı Pazarlama Karması Modeli kavramını açıklamaktır. Bu amaç doğrultusunda oluşturulmuş olan modellerden biri Knime platformu üzerinde hazırlanmıştır. Bu platformda veriyi okuma, veri görselleştirme, veri ön işleme, modelleme ve değerlendirme işlemleri yapılmıştır.

3.2.1 Veri Kümesinin Açıklanması

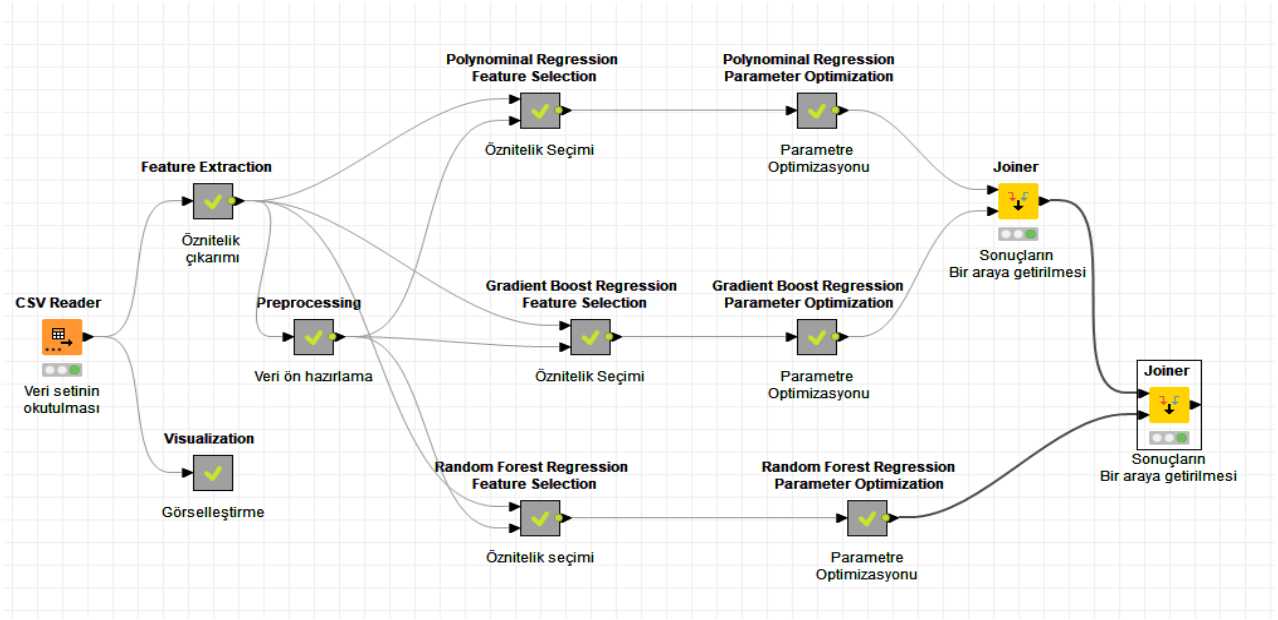
- Veri kümesine Kaggle platformunda rastlanılmıştır [9].
- Veri kümesiyle Kaggle platformunda yapılmış sadece bir proje bulunmaktadır. Bu proje de veri kümesiyle uçtan uca analiz yapılmıştır. Bu analiz veri kümesi hakkında ilk izlenimlerimizi edinmemizi sağlamaktadır.
- Veri kümesinin bir virgülle ayrılmış değerler "coma separatted values, csv" dosyasıdır.
- Veri kümesinin belirli bir markanın belirli bir ürününün 2 yıl boyunca yapılan satış değerlerini içermektedir.
- Her satır 2 yıl boyunca haftalık olarak yapılan satışlardan oluşmaktadır.
- Değişkenlerimizden bazıları ise satışlar için yapılan pazarlama faaliyetlerinin maliyet değerlerini taşımaktadır.

- Satış yapılan bu iki yılın hangi yıllar olduğu bilinmemektedir.

3.2.2. Veri Kümesindeki Değişkenlerin Açıklanması

- **NewVolSales:** Haftalık toplam satış miktarı.
- **Base_Price:** Satılan ürünün o haftaki birim satış fiyatı.
- **Radio:** Radyo kanalı üzerinden yapılan pazarlama harcamaları.
- **InStore:** Mağaza içerisinden yapılan promosyon ve pazarlama harcamaları.
- **NewspaperInserts:** Kategorik bir değişkendir. O hafta gazetelere herhangi bir harcama yapılıp yapılmadığını göstermektedir.
- **Discount:** Haftalık indirim oranı.
- **TV:** Televizyon üzerinden yapılan o haftaki pazarlama harcamaları.
- **Stout:** Ürünün dayanaklılığıdır. Burada dayanıklılıktan kasıt ürünün stoklardaki tükenme tarihi bölü ürüne yapılan ziyaret sayısıdır.
- **Website_Campaign:** O hafta hangi internet sitesi üzerinden kampanya yapıldığı gösteren kategorik değişkendir.

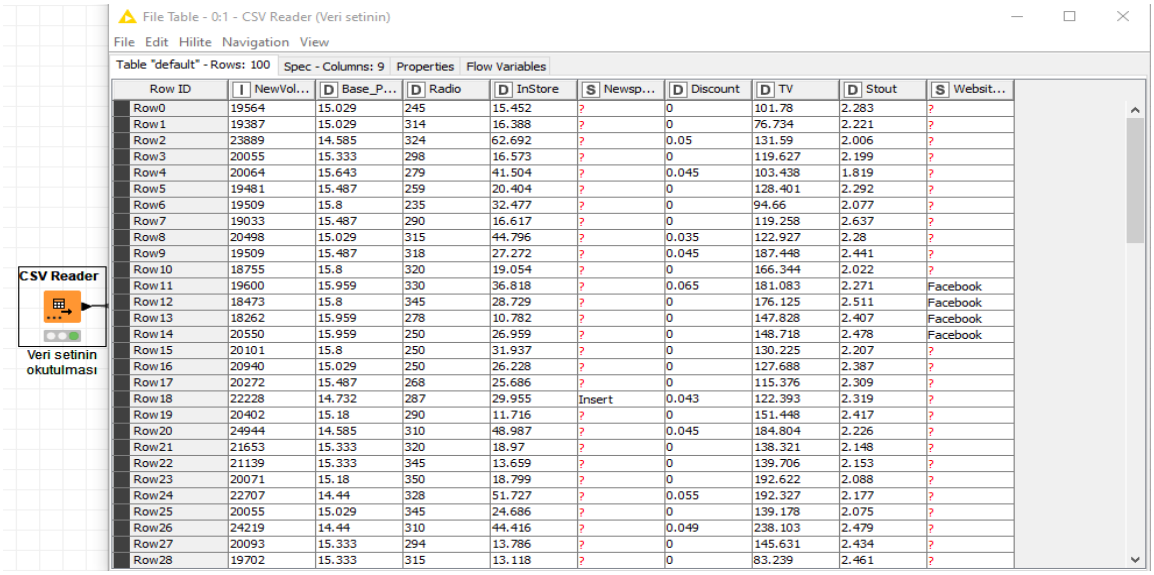
Projenin Genel Görünümü



Şekil 1 – Genel Görünüm

Projemizin genel görünüşü yukarıda görüldüğü gibidir. Görüntü olarak kalabalık gözükmemesi adına aynı işlem içerisinde sayılabilecek nodlarımızı birer Metanod içinde toplanmıştır. Makale boyunca bu MetaNod'lar teker teker açıklanacaktır.

3.2.3 Verinin Okunması



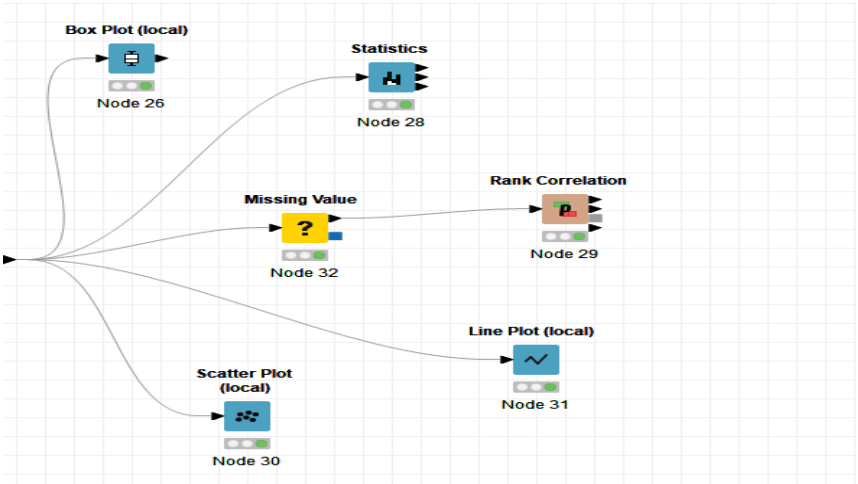
Row ID	NewVol...	Base_P...	Radio	InStore	Newsp...	Discount	TV	Stout	Websit...
Row0	19564	15.029	245	15.452	?	0	101.78	2.283	?
Row1	19387	15.029	314	16.388	?	0	76.734	2.221	?
Row2	23889	14.585	324	62.692	?	0.05	131.59	2.006	?
Row3	20055	15.333	298	16.573	?	0	119.627	2.199	?
Row4	20064	15.643	279	41.504	?	0.045	103.438	1.819	?
Row5	19481	15.487	259	20.404	?	0	128.401	2.292	?
Row6	19509	15.8	235	32.477	?	0	94.66	2.077	?
Row7	19033	15.487	290	16.617	?	0	119.258	2.637	?
Row8	20498	15.029	315	44.796	?	0.035	122.927	2.28	?
Row9	19509	15.487	318	27.272	?	0.045	187.448	2.441	?
Row10	18755	15.8	320	19.054	?	0	166.344	2.022	?
Row11	19600	15.959	330	36.818	?	0.065	181.083	2.271	Facebook
Row12	18473	15.8	345	28.729	?	0	176.125	2.511	Facebook
Row13	18262	15.959	278	10.782	?	0	147.828	2.407	Facebook
Row14	20550	15.959	250	26.959	?	0	148.718	2.478	Facebook
Row15	20101	15.8	250	31.937	?	0	130.225	2.207	?
Row16	20940	15.029	250	26.228	?	0	127.688	2.387	?
Row17	20272	15.487	268	25.686	?	0	115.376	2.309	?
Row18	22228	14.732	287	29.955	Insert	0.043	122.393	2.319	?
Row19	20402	15.18	290	11.716	?	0	151.448	2.417	?
Row20	24944	14.585	310	48.987	?	0.045	184.804	2.226	?
Row21	21653	15.333	320	18.97	?	0	138.321	2.148	?
Row22	21139	15.333	345	13.659	?	0	139.706	2.153	?
Row23	20071	15.18	350	18.799	?	0	192.622	2.088	?
Row24	22707	14.44	328	51.727	?	0.055	192.327	2.177	?
Row25	20055	15.029	345	24.686	?	0	139.178	2.075	?
Row26	24219	14.44	310	44.416	?	0.049	238.103	2.479	?
Row27	20093	15.333	294	13.786	?	0	145.631	2.434	?
Row28	19702	15.333	315	13.118	?	0	83.239	2.461	?

Şekil 2 – Verilerin Okunması

Verimizi okutmak için Knime'da bulunan CSV Reader nod'unu kullandık. Veriyle ilgili ilk gözlem olarak NewspaperInserts ve Website_Campaign kolonlarındaki kayıp veriler olmuştur.

3.2.4 Verinin Görselleştirilmesi ve Anlaşılması

Veri görselleştirilmesi veriyi anlamak için bize önemli bir fayda sağlamaktadır. Elde edilen grafikler bize veri ön işleme aşamasında yardımcı olacak ve hangi işlemleri yapmamız gerektiği konusunda bilgiler verecektir.



Şekil 3 – Görselleştirme

a) Veri Kümesinin İstatistikleri

Table "default" - Rows: 7 | Spec - Columns: 16 | Properties | Flow Variables

Row ID	S	Min	Max	Mean	Std. de...	Variance	Skewness	Kurtosis	Overall ...	No. mis...	No. NaNs	No. +oos	No. -oos	Median	Row co...	Histogram
NewVolSales	..	17,431	24,944	20,218.75	1,580.161	2,496,907.523	0.962	0.771	2,021,875	0	0	0	0	?	100	17.431 24.944
Base_Price	..	13.736	16.281	15.284	0.524	0.274	-0.644	0.168	1,528.376	0	0	0	0	?	100	14 16
Radio	..	0	399	256.69	86.995	7,568.075	-1.917	3.659	25,669	0	0	0	0	?	100	0 399
InStore	..	10.782	68.119	32.485	13.672	186.917	0.648	-0.209	3,248.493	0	0	0	0	?	100	11 68
Discount	..	0	0.091	0.021	0.027	0.001	0.753	-1.031	2.078	0	0	0	0	?	100	0 0.1
TV	V	37.656	240.292	140.263	43.533	1,895.106	-0	-0.022	14,026.309	0	0	0	0	?	100	38 240
Stout	..	1.819	3.159	2.54	0.313	0.098	0.196	-0.783	253.988	0	0	0	0	?	100	2 3

Şekil 4 – istatistikler

Şekil 4'te görüldüğü üzere bütün istatistiklerimiz bu şekildedir. Buradan sayısal değişkenlerde kategorik değişken olmadığı sonucunu çıkartabiliriz. Ayrıca '0' değeri ile başlayan değerler görülüyor. Kayıp veri durumu söz konusu olursa ortalama değer yerine medyan değeri kullanmamız daha mantıklı olacaktır.

b) Değişkenler Arasında Korelasyon Analizi

Korelasyon analizi değişkenler arası ilişkilerin yönünü ve derecesini tanımlamak için kullanılan bir istatistiksel analiz tekniğidir. İstatistik analizi uygulamalarında verilerimiz nicel veya sıralayıcı (ordinal) ölçeğe sahip olduğunda Spearman korelasyon analizine başvurabiliriz. Spearman korelasyon analizi, normal dağılım koşulunu gerektirmediği için son derece esnek bir tekniktir. Ancak parametrik olmayan tüm analizlerin bir bedeli var. Pratik istatistik analizi uygulamalarında Spearman korelasyon katsayıları bağlamında olası problemlere hazır olmamız gerekmektedir.[10]

$$\text{variance}(s^2) = \frac{\sum(x_i - \overline{x})^2}{N-1} = \frac{\sum(x_i - \overline{x})(x_i - \overline{x})}{N-1}$$

Formül 1 – Kolerasyon Formülü

Kolaresyon formülü Formül 1'de görüldüğü gibidir. Knime programında bu analizi bir nod ile hesaplayabilmekteyiz.

Correlation matrix - 0:32:29 - Rank Correlation

File Edit Hilite Navigation View

Table "Correlation values" - Rows: 9 | Spec - Columns: 9 | Properties | Flow Variables

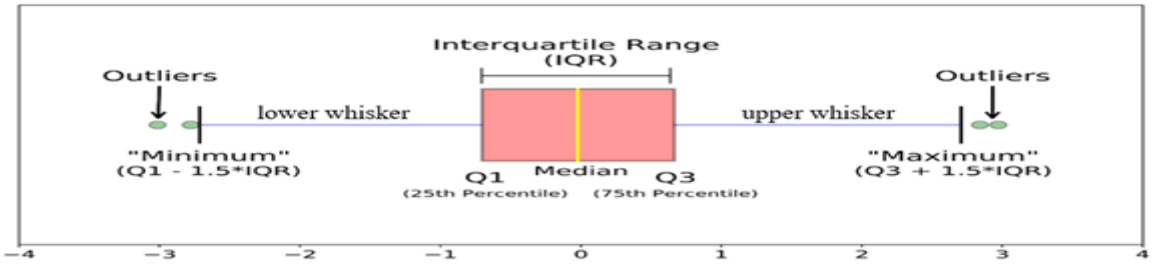
Row ID	NewVol...	Base_...	Radio	InStore	NewspaperInserts	Discount	TV	Stout	Website_Ca...
NewVolSales	1.0	-0.6643289...	0.17499419...	0.39912630...	0.0714778487055794	0.45660827549...	0.18238333...	-0.38234852895095...	0.147313555428...
Base_Price	-0.6643289...	1.0	0.04065631...	-0.23179917...	-0.11800249560349...	-0.2478324569...	0.05089318...	1.447199649560216...	-0.026618991957...
Radio	0.17499419...	0.0406563...	1.0	-0.01905500...	0.036496991289570 81	0.12889461180...	0.16188645...	-0.3352191206849753	0.058149203796...
InStore	0.39912630...	-0.2317991...	-0.0190550...	1.0	0.013128505676505...	0.67218329816...	0.02334233...	0.0422682268226823	-0.057734944595...
NewspaperIn...	0.07147784...	-0.1180024...	0.03649699...	0.01312850...	1.0	0.00497731285...	0.03355062...	-0.06272508267663...	-0.101598763903...
Discount	0.45660827...	-0.2478324...	0.12889461...	0.67218329...	0.00497731285151757	1.0	0.12002249...	-0.06190174143958...	0.046742504488...
TV	0.18238333...	0.0508931...	0.16188645...	0.02334233...	0.03355062561773697	0.12002249448...	1.0	-0.1263606360636064	-0.115927629185...
Stout	-0.3823485...	1.4471996...	-0.3352191...	0.04226822...	-0.06272508267663...	-0.0619017414...	-0.1263606...	1.0	-0.144566231485...
Website_Cam...	0.14731355...	-0.0266189...	0.05814920...	-0.05773494...	-0.10159876390304...	0.04674250448...	-0.1159276...	-0.14456623148599...	1.0

Şekil 5 - Korelasyon

Korelasyon ilişkileri Şekil 5'te görüldüğü gibidir. Base_Price ile NewVolSales değişkenleri arasında %66'lık bir negatif korelasyon ilişkisi gözlemlenmiştir.

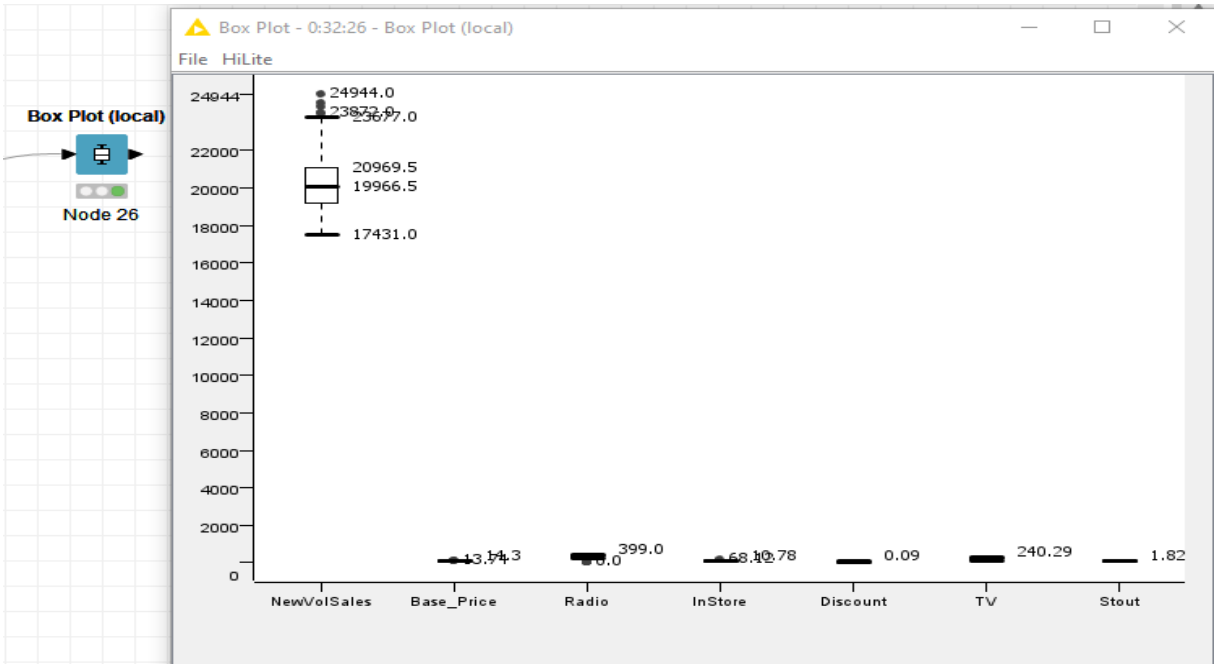
c) Aykırı Değer Tespiti

Veri kümesi karakteristiğine uygun olmayan ve veri kümesindeki diğer gözlemlerden belirgin bir şekilde sapan gözlemler aykırı gözlem olarak adlandırılmaktadır. Veri kümesinin gözlenen veya teorik olarak tahmin edilen örüntüsünün aksine farklı örüntüler içeren aykırı gözlemler, verinin üretilme veya ölçüm sürecinden kaynaklandığı gibi, örneklem hatasından veya örneklem dağılımına ilişkin varsayımlardan da kaynaklanmaktadır. Veri kümesinin kalitesini olumsuz olarak etkileyen aykırı değerlerin veri kümesinden çıkartılması, gerçekleştirilecek istatistiki analizlerin sonuçlarının güvenilirliği açısından önemli olacağı için, veri kümesinde bulunan aykırı değerlerin doğru olarak tespit edilerek veri kümesinden çıkartılması, normalize edilmesi veya dönüştürülerek homojenliğin sağlanması önerilmektedir.[11]



Formül 2 – Aykırı Değer Tespiti

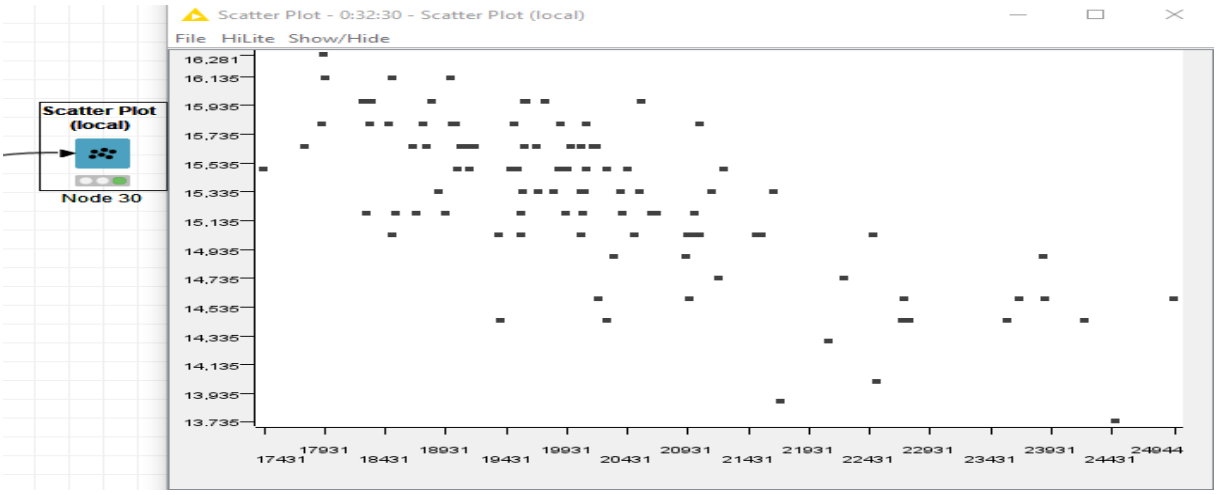
Aykırı değerlerin tespiti Formül 2'de görüldüğü gibidir. Knime programında bu analizi bir nod ile hesaplayabilmekteyiz.



Şekil 6 – Aykırı Değerler 1

Şekil 6'da görüldüğü üzere numerik değişkenlerimizin bazılarında aykırı değerler bulunuyor. Gerekli işlemler veri ön hazırlık aşamasında yapılacaktır.

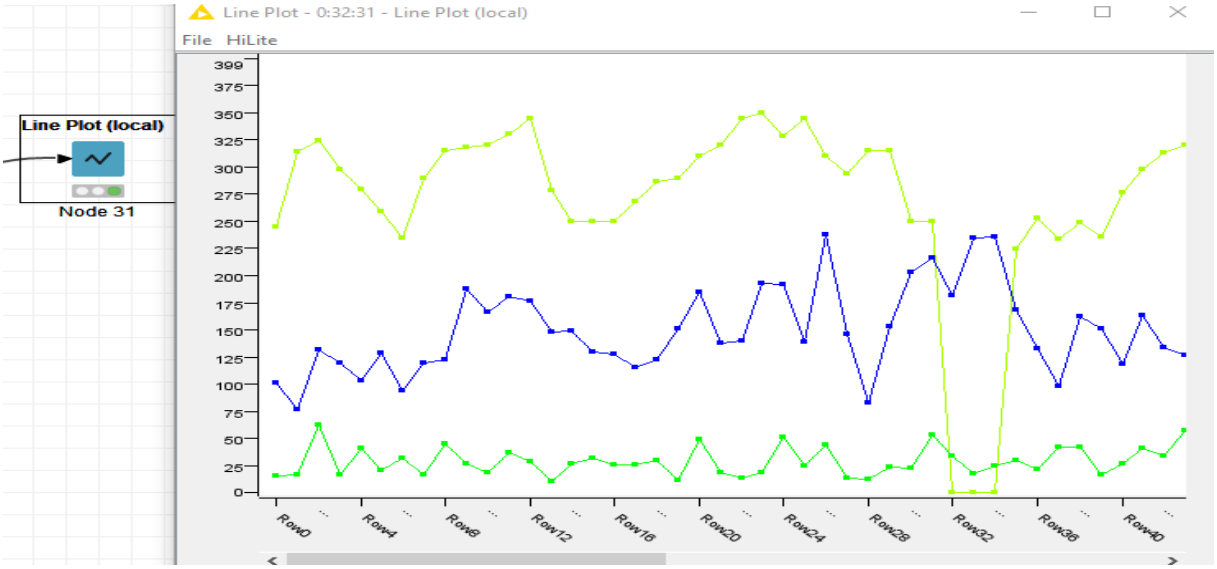
d) Dağılım Grafiği



Şekil 7 – Dağılım Grafiği 1

Şekil 7'de x eksenimiz hedef değişkenimiz olan NewVolSales değişkeni, y ekseni ise Haftalık birim fiyatı belirten Base_Price değişkenimizdir. Bu durumda bu şekilden haftalık birim fiyatın düşük olduğu haftalarda toplam satışın yüksek olduğunu, haftalık birim satış fiyatının yüksek olduğu haftalarda ise toplam satışın az olduğunu gözlemleyebiliriz. Daha önce gördüğümüz bu iki değişken arasındaki negatif korelasyon ilişkisi bu şekilde açıklanabilir.

e) Çizgi Grafiği



Şekil 8 – Çizgi Grafiği

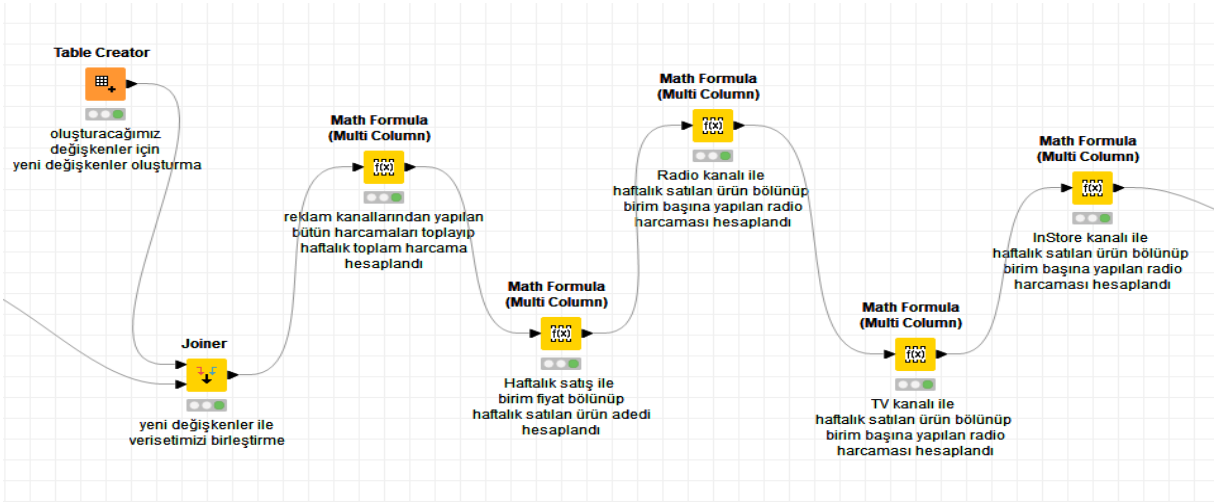
Çizgi grafiğine sadece reklam kanallarını içeren değişkenler eklenmiştir. Bu sayede 3 farklı kanala ne kadar harcama yapıldığı aynı anda gözlemlenmiştir. Sarı çizgiler Radio harcamalarını, mavi çizgiler TV harcamalarını ve yeşil çizgiler ise Instore harcamalarını göstermektedir. En çok harcama yapılan kanal önce Radio sonra TV ve en az harcama yapılan kanal ise Instore harcamalarıdır.

3.2.5 Öznitelik Mühendisliği (Feature Engineering)

Öznitelik mühendisliği nedir ve neden ihtiyacımız vardır. Basitçe anlatmak gerekirse bütün makine öğrenmesi algoritmaları verileri girdi olarak alıp çıktıları bilgi olarak bize geri vermektedir. Girdi için kullanılan veriler genellikle kolonlar şeklindedir ve bazen bu veri kümeleri bizim istediğimiz sonuçları çıkarmakta güçlük çekebilmektedir. Bu durumlarda algoritma daha fazla veri veya değişkene ihtiyaç duyar. Öznitelik seçimine burada ihtiyaç duyulmaktadır. Veri kümesinde halı hazırda bulunan verileri kaybetmeden onlardan yeni değişkenler oluşturarak modelimizin öğrenmesini güçlendirmekteyiz. Öznitelik mühendisliğinin iki ana hedefi bulunmakdatır. Veri kümesine yeni girdiler eklemek ve eklenen yeni girdilerle algoritma ihtiyaçlarını karşılamak ve makine öğrenmesi modellerinin performansını yükseltmektir.[12]

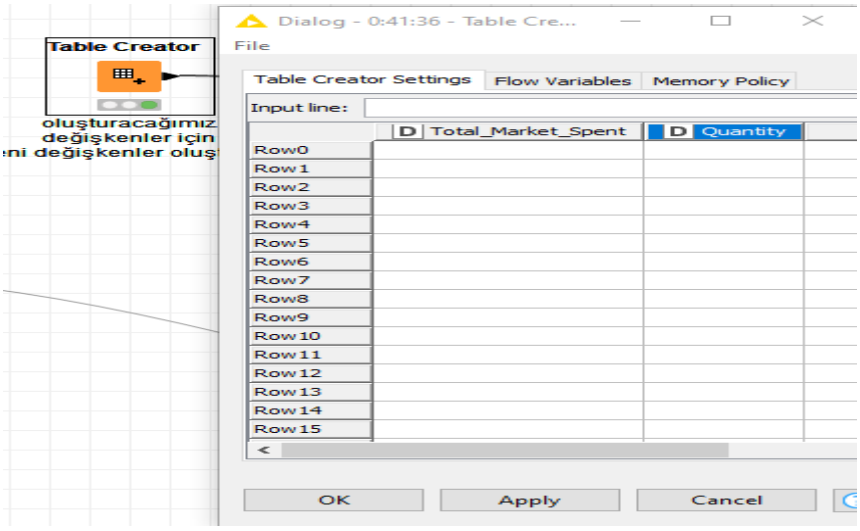
a) Feature Engineering Metanod'u

Verimizin daha iyi eğitebilmek için oluşturulan değişkenlerden oluşan Metanod'dur.



Şekil 9 – Öznitelik Mühendisliği

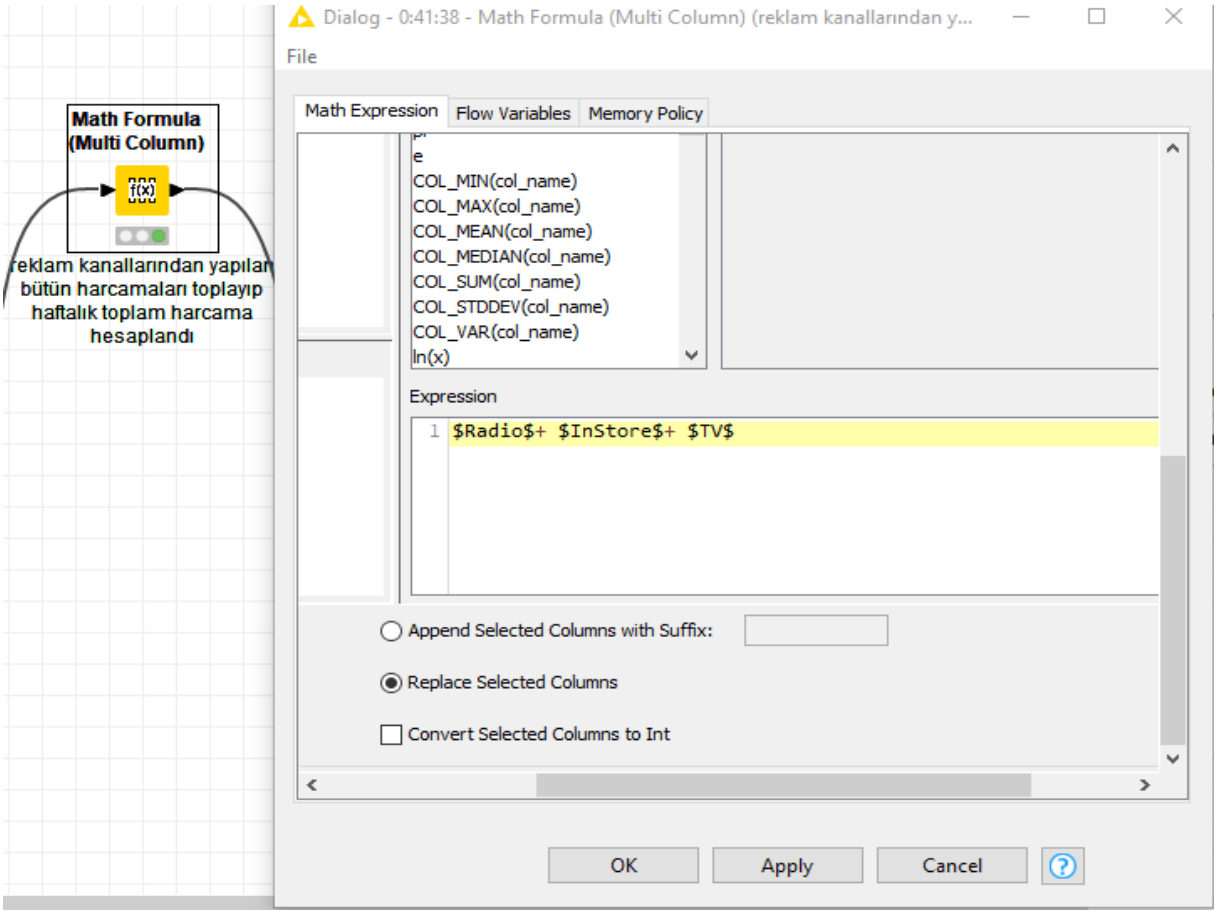
b) Öznitelik Mühendisliği ile Yeni Değişkenler Oluşturulması



Şekil 10 – Değişken Oluşturma

Biz burada veri kümemize ekleyeceğimiz yeni değişkenler için yeni kolonlar oluşturduk.

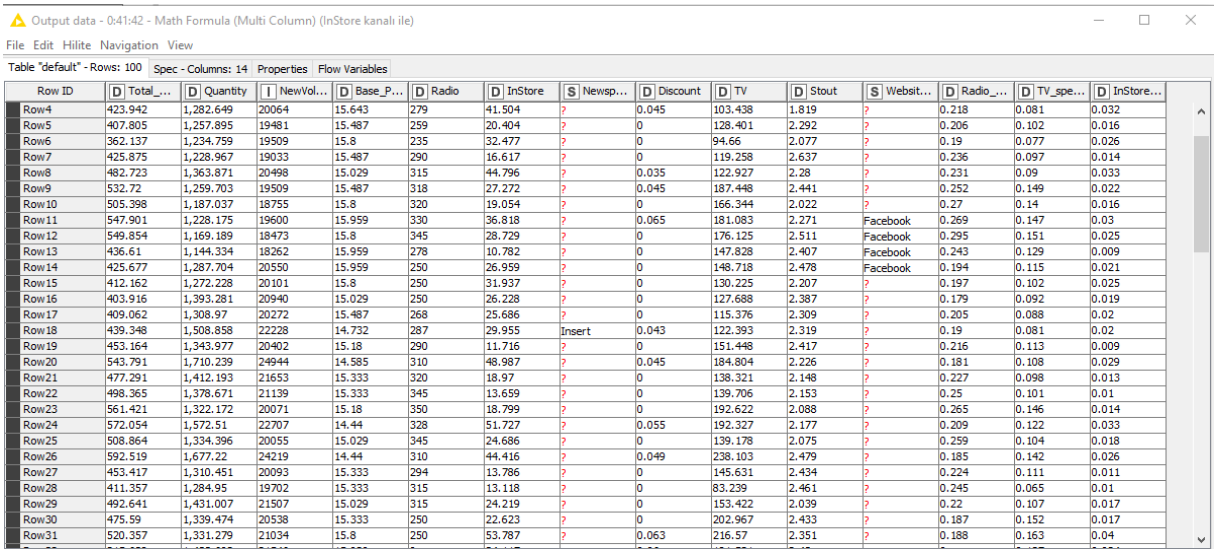
c) Oluşturulan Yeni Değişkenlerin Değerlerinin Hesaplanması



Şekil 11 – Yeni Değişken Förmülü

Şekil 11'de görüldüğü gibi yeni değişkenler oluşturmak için Math Formula Nod'unun Expression alanına işlem yapmak istediğiniz kolonları ve yapılmasını istediğiniz işlemleri belirterek yeni değişkenler oluşturmamız mümkündür.

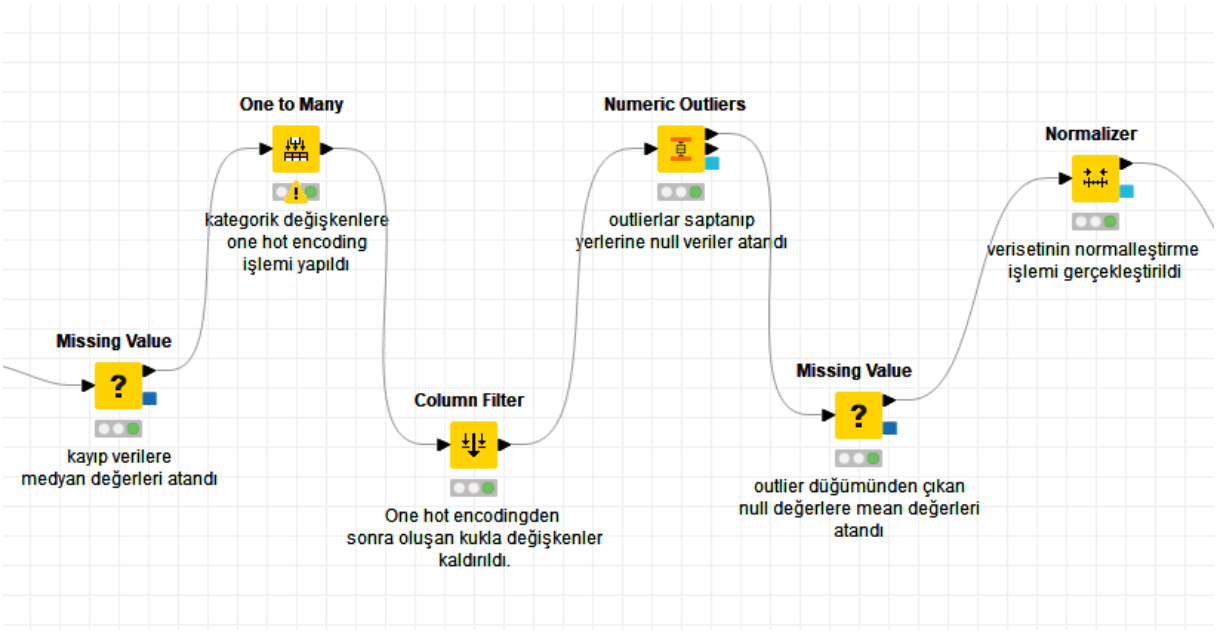
3.2.6 Veri Ön İşleme İşlemleri



Row ID	Total_...	Quantity	NewVol...	Base_P...	Radio	InStore	Newsp...	Discount	TV	Stout	Websit...	Radio_...	TV_spe...	InStore...
Row4	423.942	1,282.649	20064	15.643	279	41.504	?	0.045	103.438	1.819	?	0.218	0.081	0.032
Row5	407.805	1,257.895	19481	15.487	259	20.404	?	0	128.401	2.292	?	0.206	0.102	0.016
Row6	362.137	1,234.759	19509	15.8	235	32.477	?	0	94.66	2.077	?	0.19	0.077	0.026
Row7	425.875	1,228.967	19033	15.487	290	16.617	?	0	119.258	2.637	?	0.236	0.097	0.014
Row8	482.723	1,363.871	20498	15.029	315	44.796	?	0.035	122.927	2.28	?	0.231	0.09	0.033
Row9	532.72	1,259.703	19509	15.487	318	27.272	?	0.045	187.448	2.441	?	0.252	0.149	0.022
Row10	505.398	1,187.037	18755	15.8	320	19.054	?	0	166.344	2.022	?	0.27	0.14	0.016
Row11	547.901	1,228.175	19600	15.959	330	36.818	?	0.065	181.083	2.271	Facebook	0.269	0.147	0.03
Row12	549.854	1,169.189	18473	15.8	345	28.729	?	0	176.125	2.511	Facebook	0.295	0.151	0.025
Row13	436.61	1,144.334	18262	15.959	278	10.782	?	0	147.828	2.407	Facebook	0.243	0.129	0.009
Row14	425.677	1,287.704	20550	15.959	250	26.959	?	0	148.718	2.478	Facebook	0.194	0.115	0.021
Row15	412.162	1,272.228	20101	15.8	250	31.937	?	0	130.225	2.207	?	0.197	0.102	0.025
Row16	403.916	1,393.281	20940	15.029	250	26.228	?	0	127.688	2.387	?	0.179	0.092	0.019
Row17	409.062	1,308.97	20272	15.487	268	25.686	?	0	115.376	2.309	?	0.205	0.088	0.02
Row18	439.348	1,508.858	22228	14.732	287	29.955	Insert	0.043	122.393	2.319	?	0.19	0.081	0.02
Row19	453.164	1,343.977	20402	15.18	290	11.716	?	0	151.448	2.417	?	0.216	0.113	0.009
Row20	543.791	1,710.239	24944	14.585	310	48.987	?	0.045	184.804	2.226	?	0.181	0.108	0.029
Row21	477.291	1,412.193	21653	15.333	320	18.97	?	0	138.321	2.148	?	0.227	0.098	0.013
Row22	498.365	1,378.671	21139	15.333	345	13.659	?	0	139.706	2.153	?	0.25	0.101	0.01
Row23	561.421	1,322.172	20071	15.18	350	18.799	?	0	192.622	2.088	?	0.265	0.146	0.014
Row24	572.054	1,572.51	22707	14.44	328	51.727	?	0.055	192.327	2.177	?	0.209	0.122	0.033
Row25	508.864	1,334.396	20055	15.029	345	24.686	?	0	139.178	2.075	?	0.259	0.104	0.018
Row26	592.519	1,677.22	24219	14.44	310	44.416	?	0.049	238.103	2.479	?	0.185	0.142	0.026
Row27	453.417	1,310.451	20093	15.333	294	13.786	?	0	145.631	2.434	?	0.224	0.111	0.011
Row28	411.357	1,284.95	19702	15.333	315	13.118	?	0	83.239	2.461	?	0.245	0.065	0.01
Row29	492.641	1,431.007	21507	15.029	315	24.219	?	0	153.422	2.039	?	0.22	0.107	0.017
Row30	475.59	1,339.474	20538	15.333	250	22.623	?	0	202.967	2.433	?	0.187	0.152	0.017
Row31	520.357	1,331.279	21034	15.8	250	53.787	?	0.063	216.57	2.351	?	0.188	0.163	0.04

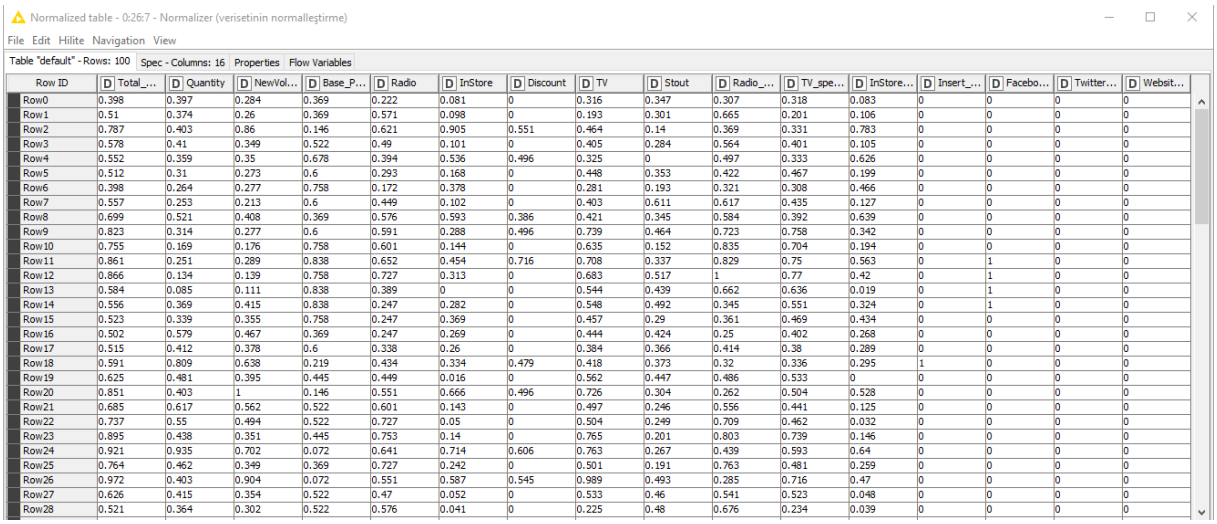
Şekil 12 – Veri Ön Hazırlık Öncesi

Veri Ön Hazırlık işlemlerinden önce veri kümemiz şekil 12'deki gibi gözükmektedir.



Şekil 13 – Veri Ön Hazırlık

Şekil 13'teki işlemlerin ardından veri kümemiz eğitim için modele hazır hale getirildi.



Normalized table - 0:26:7 - Normalizer (verisetinin normalleştirme)

File Edit Hilite Navigation View

Table "default" - Rows: 100 | Spec - Columns: 16 | Properties | Flow Variables

Row ID	Total_...	Quantity	NewVol...	Base_P...	Radio	InStore	Discount	TV	Stout	Radio_...	TV_spe...	InStore...	Insert_...	Facebo...	Twitter...	Websit...
Row0	0.398	0.397	0.284	0.369	0.222	0.081	0	0.316	0.347	0.307	0.318	0.083	0	0	0	0
Row1	0.51	0.374	0.26	0.369	0.571	0.098	0	0.193	0.301	0.665	0.201	0.106	0	0	0	0
Row2	0.787	0.403	0.86	0.146	0.621	0.905	0.551	0.464	0.14	0.369	0.331	0.783	0	0	0	0
Row3	0.578	0.41	0.349	0.522	0.49	0.101	0	0.405	0.284	0.564	0.401	0.105	0	0	0	0
Row4	0.552	0.359	0.35	0.678	0.394	0.536	0.496	0.325	0	0.497	0.333	0.626	0	0	0	0
Row5	0.512	0.31	0.273	0.6	0.293	0.168	0	0.448	0.353	0.422	0.467	0.199	0	0	0	0
Row6	0.398	0.264	0.277	0.758	0.172	0.378	0	0.281	0.193	0.321	0.308	0.466	0	0	0	0
Row7	0.557	0.253	0.213	0.6	0.449	0.102	0	0.403	0.611	0.617	0.435	0.127	0	0	0	0
Row8	0.699	0.521	0.408	0.369	0.576	0.593	0.386	0.421	0.345	0.584	0.392	0.639	0	0	0	0
Row9	0.823	0.314	0.277	0.6	0.591	0.288	0.496	0.739	0.464	0.723	0.758	0.342	0	0	0	0
Row10	0.755	0.169	0.176	0.758	0.601	0.144	0	0.635	0.152	0.835	0.704	0.194	0	0	0	0
Row11	0.861	0.251	0.289	0.838	0.652	0.454	0.716	0.708	0.337	0.829	0.75	0.563	0	1	0	0
Row12	0.866	0.134	0.139	0.758	0.727	0.313	0	0.683	0.517	1	0.77	0.42	0	1	0	0
Row13	0.584	0.085	0.111	0.838	0.389	0	0	0.544	0.439	0.662	0.636	0.019	0	1	0	0
Row14	0.556	0.369	0.415	0.838	0.247	0.282	0	0.548	0.492	0.345	0.551	0.324	0	1	0	0
Row15	0.523	0.339	0.355	0.758	0.247	0.369	0	0.457	0.29	0.361	0.469	0.434	0	0	0	0
Row16	0.502	0.579	0.467	0.369	0.247	0.269	0	0.444	0.424	0.25	0.402	0.268	0	0	0	0
Row17	0.515	0.412	0.378	0.6	0.338	0.26	0	0.384	0.366	0.414	0.38	0.289	0	0	0	0
Row18	0.591	0.809	0.638	0.219	0.434	0.334	0.479	0.418	0.373	0.32	0.336	0.295	1	0	0	0
Row19	0.625	0.481	0.395	0.445	0.449	0.016	0	0.562	0.447	0.486	0.533	0	0	0	0	0
Row20	0.851	0.403	1	0.146	0.551	0.666	0.496	0.726	0.304	0.262	0.504	0.528	0	0	0	0
Row21	0.685	0.617	0.562	0.522	0.601	0.143	0	0.497	0.246	0.556	0.441	0.125	0	0	0	0
Row22	0.737	0.55	0.494	0.522	0.727	0.05	0	0.504	0.249	0.709	0.462	0.032	0	0	0	0
Row23	0.895	0.438	0.351	0.445	0.753	0.14	0	0.765	0.201	0.803	0.739	0.146	0	0	0	0
Row24	0.921	0.935	0.702	0.072	0.641	0.714	0.606	0.763	0.267	0.439	0.593	0.64	0	0	0	0
Row25	0.764	0.462	0.349	0.369	0.727	0.242	0	0.501	0.191	0.763	0.481	0.259	0	0	0	0
Row26	0.972	0.403	0.904	0.072	0.551	0.587	0.545	0.989	0.493	0.285	0.716	0.47	0	0	0	0
Row27	0.626	0.415	0.354	0.522	0.47	0.052	0	0.533	0.46	0.541	0.523	0.048	0	0	0	0
Row28	0.521	0.364	0.302	0.522	0.576	0.041	0	0.225	0.48	0.676	0.234	0.039	0	0	0	0

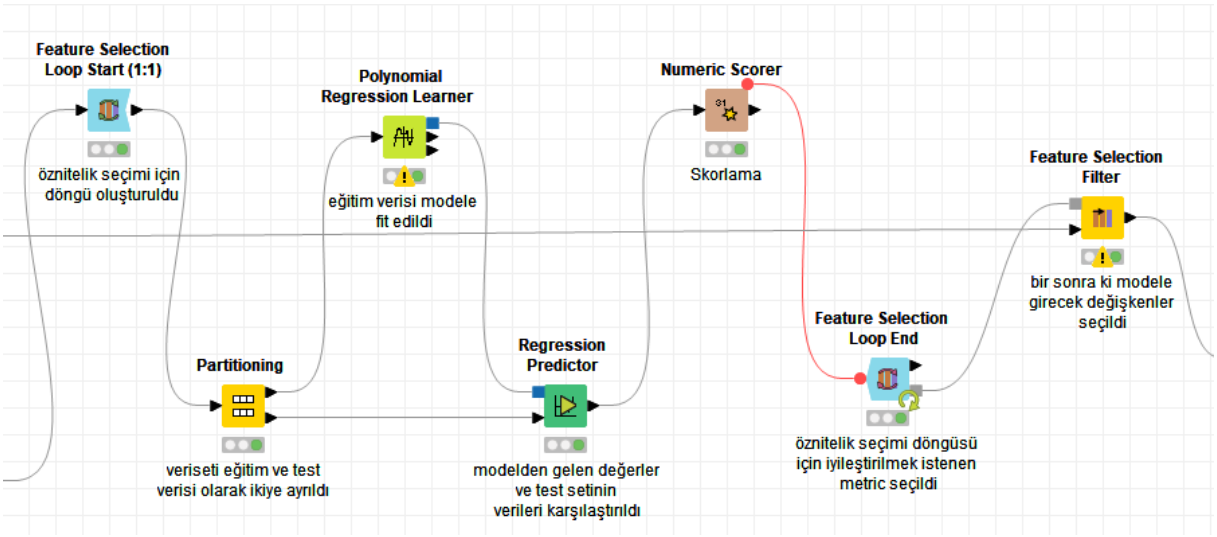
Şekil 14 – Veri Ön Hazırlık Sonrası

Şekil 14'te ile Şekil 12'ye bakarak yapılan veri ön hazırlık işlemlerini gözle görebilmekteyiz.

3.2.7 Polinominal Regresyon

Polinominal Regresyon projemizde kullandığımız 3 makine öğrenmesinden birincisidir. Önce Öznitelik Seçimi sonra ise Parametre optimizasyonu yaparak bu makine öğrenmesi algoritmasında verim elde etmeye çalışacağız.

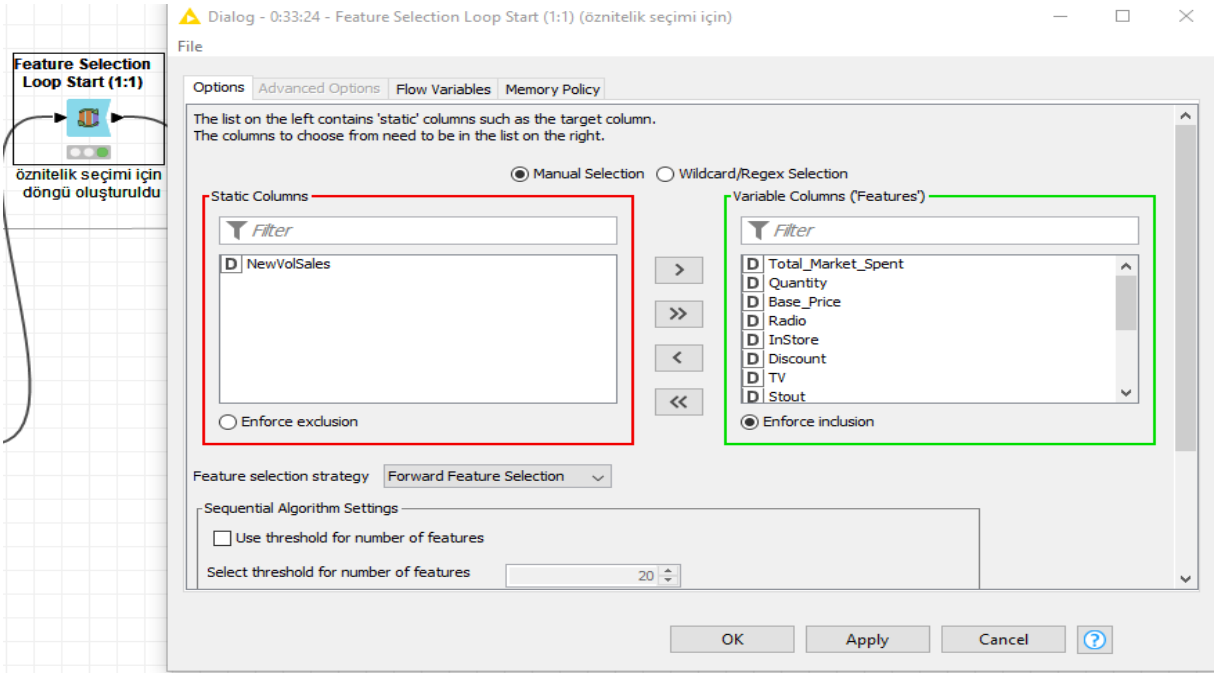
3.2.7.1 Polynomial Regression Feature Selection MetaNod'u



Şekil 15 – Polinominal Regresyon Öznitelik Seçimi

Feature Selection MetaNod'u şekil 15'te görüldüğü gibidir.

a) Öznitelik Seçimi Döngüsü Başlangıcı

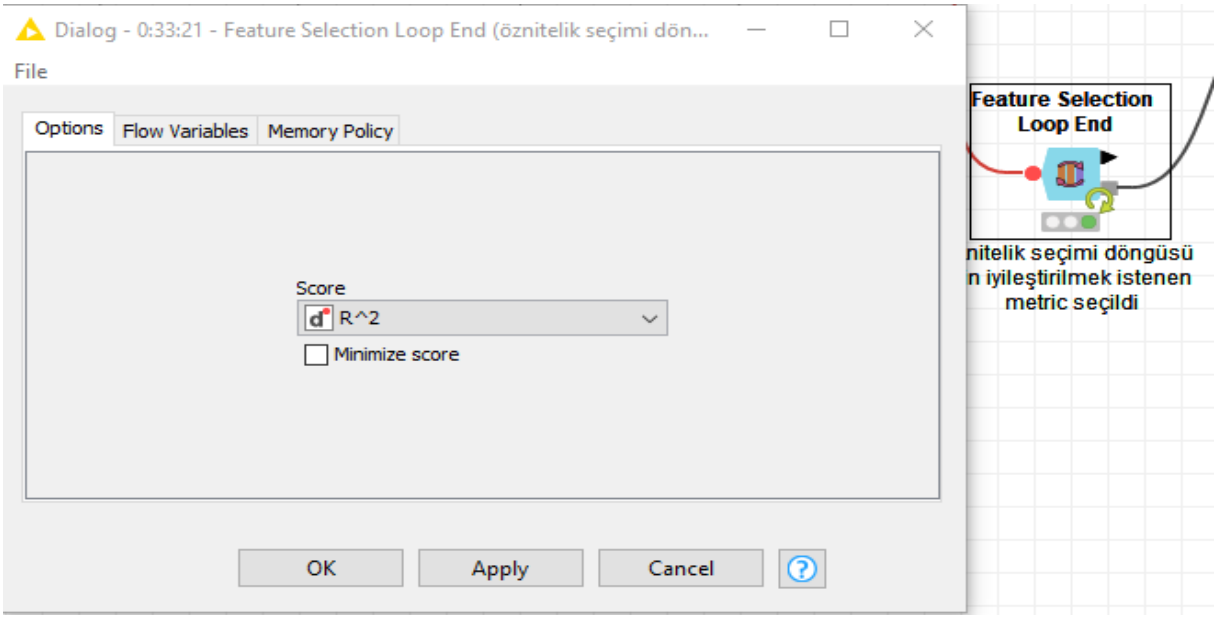


Şekil 16 – Öznitelik Seçim Döngüsü Başlangıcı 1

Feature Selection döngüsünü çalıştırmak için hedef değişkenimizi Static Columns sekmesine atıyoruz ve Feature Selection Strategy seçeneklerinden Forward Feature Selection'ı seçiyoruz.

b) Öznitelik Seçimi Döngüsü Bitişi

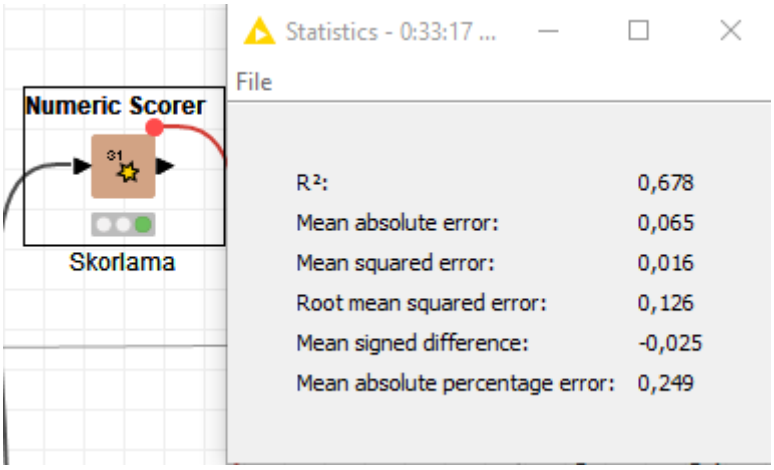
Makine öğrenmesi algoritmasını çalıştırmadan önce döngünün biteceği Nod'da maksimize veya minimize etmek istediğimiz metriği seçmemiz gerekmektedir.



Şekil 17 – Öznitelik Seçim Döngüsü Sonu 1

Öznitelik seçimi döngüsünün sonunda R^2 metriğinin maksimize edilmesini tercih ettik bizim başarı metriğimiz R^2metriği olacaktır.

c) Makine Öğrenmesi Performansının Skorlanması



Şekil 18 – Skorlama 1

Polinominal Regresyon algoritmasından aldığımız ilk sonuçlar bu şekildedir. Elde Edilen sonuçların iyileştirilmeye ihtiyaç duyduğunu görmekteyiz.

Şekil 18' de elde ettiğimiz değerlere metrikler denmektedir. Bu metrikler kurmuş olduğumuz modelin performansının nasıl olduğu hakkında bize bilgiler vermektedir. Bu araştırma kapsamında kurulan modeller için R^2 metriğini hedef metriği olarak ele alacağız. Bunun sebebi ise R^2 değerinin genel olarak 0 ile 1 arasında geldiği için anlaması ve yorumlanması daha kolay olduğunu düşünülebilir.

R^2 skoru nedir ?

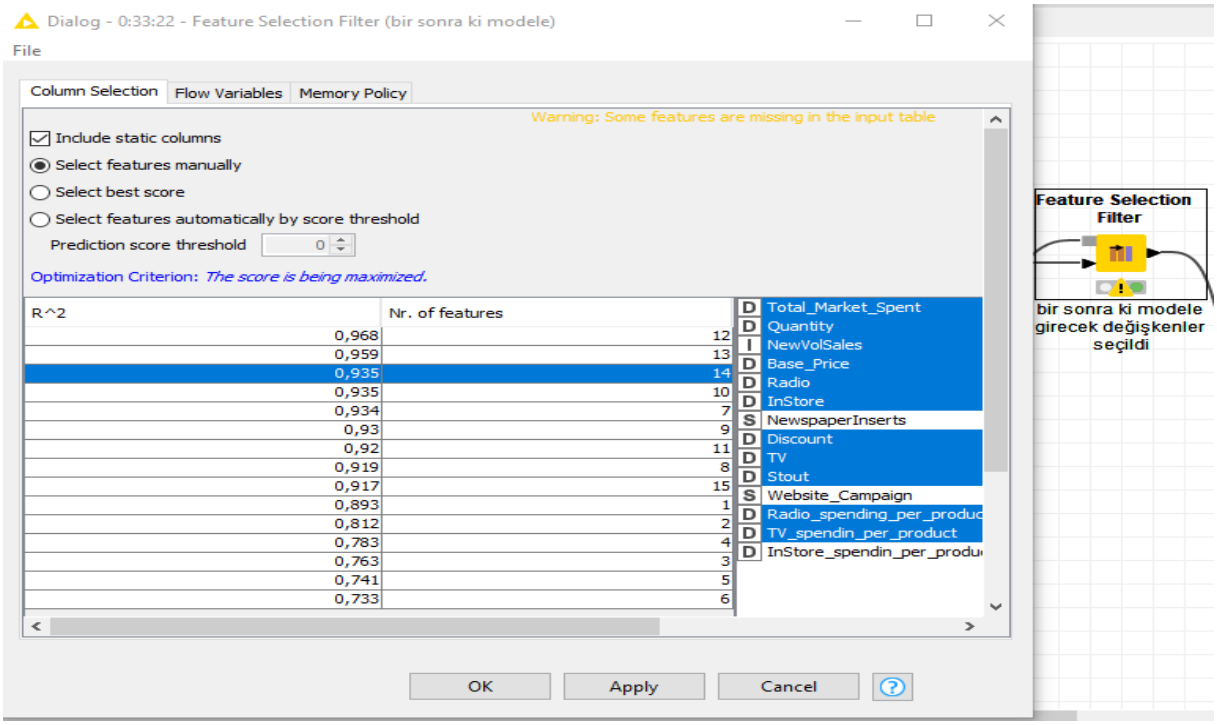
R^2, verilerin yerleştirilmiş regresyon hattına ne kadar yakın olduğunun istatistiksel bir ölçüsüdür. Ayrıca belirleme katsayısı veya çoklu regresyon için çoklu belirleme katsayısı olarak da bilinir. Daha basit bir dilde söylemek gerekirse R^2, doğrusal regresyon modelleri için uygunluk ölçüsüdür.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Formül 3 – R Kare Formülü

Burada, SSres, model tarafından açıklanan Varyans olarak da bilinen Karesel Regresyonun Toplamıdır. SStot, Toplam Kareler Toplamıdır. "yi" Gerçek Gözlemdir. "yi_cap" Öngörülen Gözlemdir. "y_bar" Gerçek değerin ortalamasıdır [13].

d) Öznitelik Seçimi Döngüsü Sonrası Değişken Seçimi



Şekil 19 – Öznitelik Seçim Filtresi

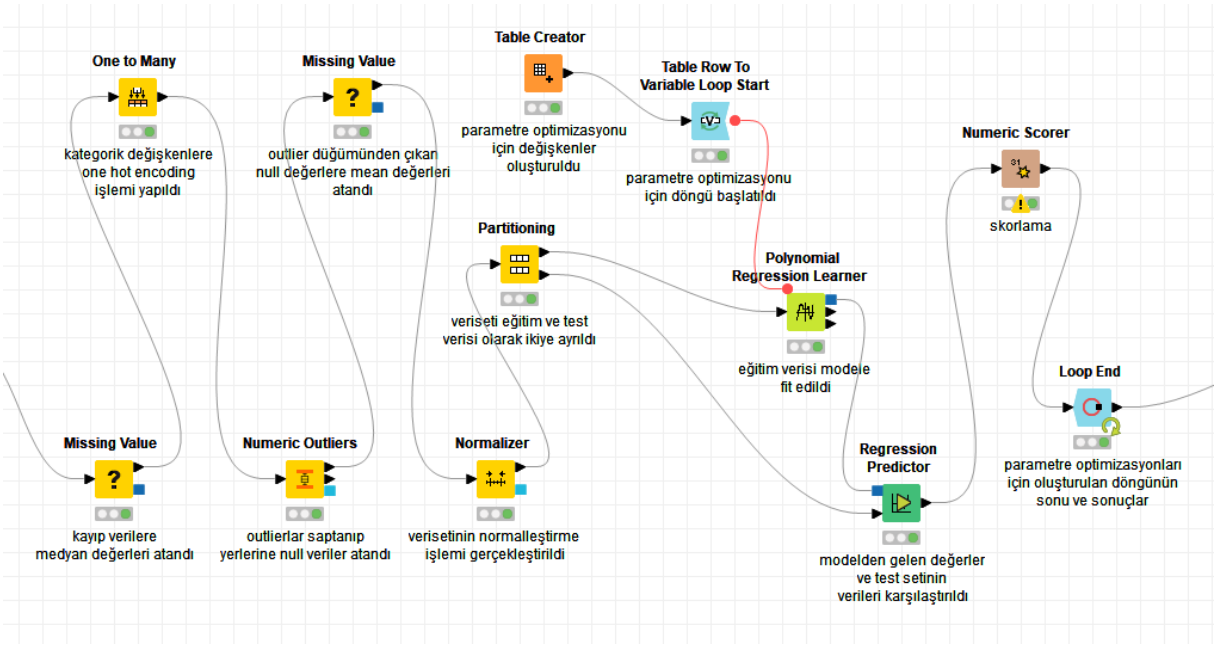
Şekil 19'da da görüldüğü üzere Feature Selection Filter Nod'unda değişkenlerin R^2 metrik değerleri, bu değerlerin ortaya çıkmasını sağlayan değişkenler ve kaç değişken kullanıldığı gösterilmektedir. Bu değişkenler arasından istediğimiz değişken kombinasyonlarından birini seçip yeni kuracağımız modele yönlendirebiliriz.

3.2.7.2 Polynomial Regression Parameter Optimization Metanod'u

Günümüzde optimizasyon işlemleri, mühendislik tasarımlarında kullanılan önemli bir yöntemdir. Optimizasyon yöntemlerini kullanmanın önemli avantajları olduğu bilinmektedir. Gerçek sistem tasarımı için, geleneksel yöntemler deneysel formülleri kullanarak yalnızca yapılabilirliğini esas alan çözümler içerirken, optimizasyon yöntemleri gerçekçi fiziksel modellere dayalı optimum sonuçlar arar. Mühendislik tasarımında kullanılan optimizasyon yöntemleri çok sayıda işlem yapmayı gerektirir. Çünkü, optimizasyon yöntemlerinin çözmeye çalıştıkları karmaşık fonksiyonların farklı potansiyel çözümler için irdelenmesi gerekmektedir [14].

Bu çalışmada üzerinde durulan Parametre ayarlama problemi bir optimizasyon problemi olup, gerçek zamanlı optimizasyon modelleme uygulamalarını içermektedir. Dinamik verilerin kullanıldığı büyük parametre uzaylarına sahip modellerin parametrelerinin ayarlanması, sistemlerin tasarımı ve kontrolü için önemlidir. Parametre ayarlama problemlerinde çözüme ulaşmak için farklı yaklaşımlardan yararlanılabilinir. Ancak bütün yaklaşımlarda amaç, deneysel ve modelden alınan veriler arasındaki farkı en aza indirgemektir. Model parametrelerinin belirlenmesinde uygun deneysel verilerin elde edilmesi

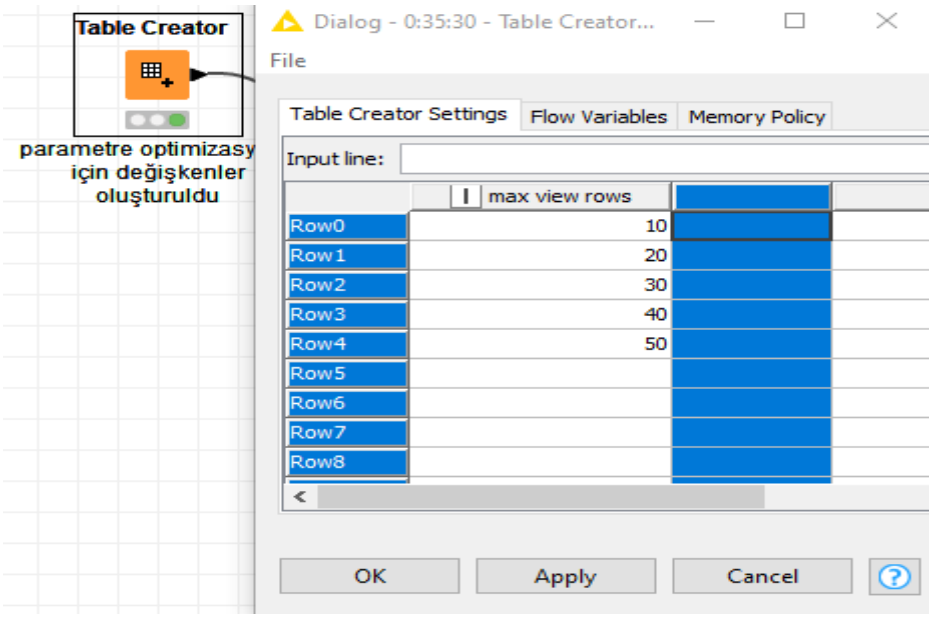
önemlidir. Modelin süreç gösterim kabiliyeti, modelde yer alan parametrelerin doğru bir şekilde belirtildiği gerçeği ile büyük oranda orantılıdır [15].



Şekil 20- Parameter Optimizasyonu 1

Şekil 20'de görüldüğü üzere bu MetaNod'da bulunan ilk 5 nod daha önce yaptığımız veri ön hazırlık işlemleridir.

a) Parametre Optimizasyonu İçin Değişken Oluşturma

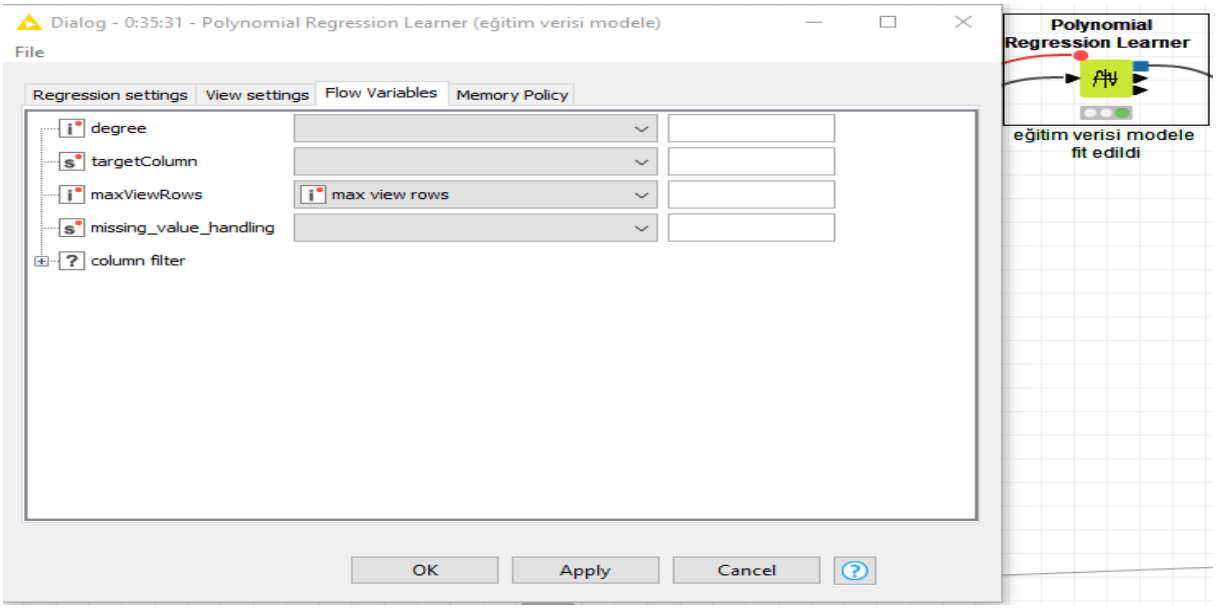


Şekil 21 – Parametre Oluşturma

Algoritmamızın içine yerleştirmek istediğimiz yeni parametremiz Şekil 21'de görüldüğü gibidir. Döngü bu değerleri teker teker deneyip sonuçlarını bize getirecektir.

b) Parametrenin Algoritmaya Yerleştirilmesi

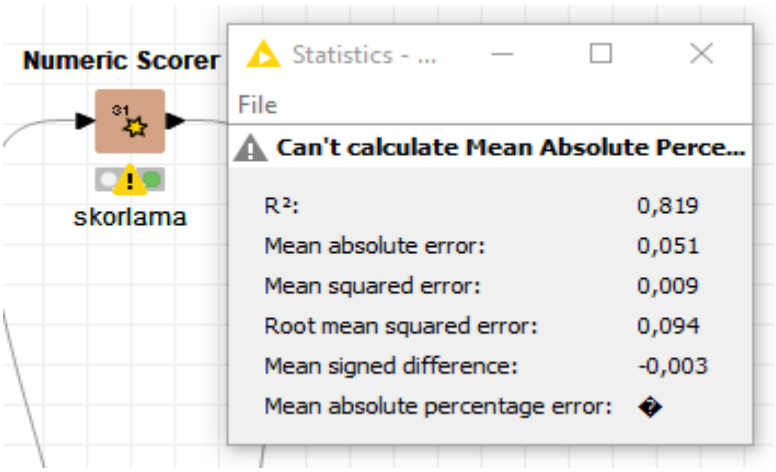
Makine öğrenmesi algoritmamızın parametrelerini optimize edebilmek için oluşturduğumuz değerleri modelimize eklememiz gerekmektedir.



Şekil 22 – Parametre Yerleştirme 1

Polynomial Regressin Learner Nod'unda bulunan Flow Variables sekmesindeki hazır olarak gelen değişkenlerle daha önceden hazırladığımız değişkenleri değiştiriyoruz.

a) Makine Öğrenmesi Performansının Skorlanması.



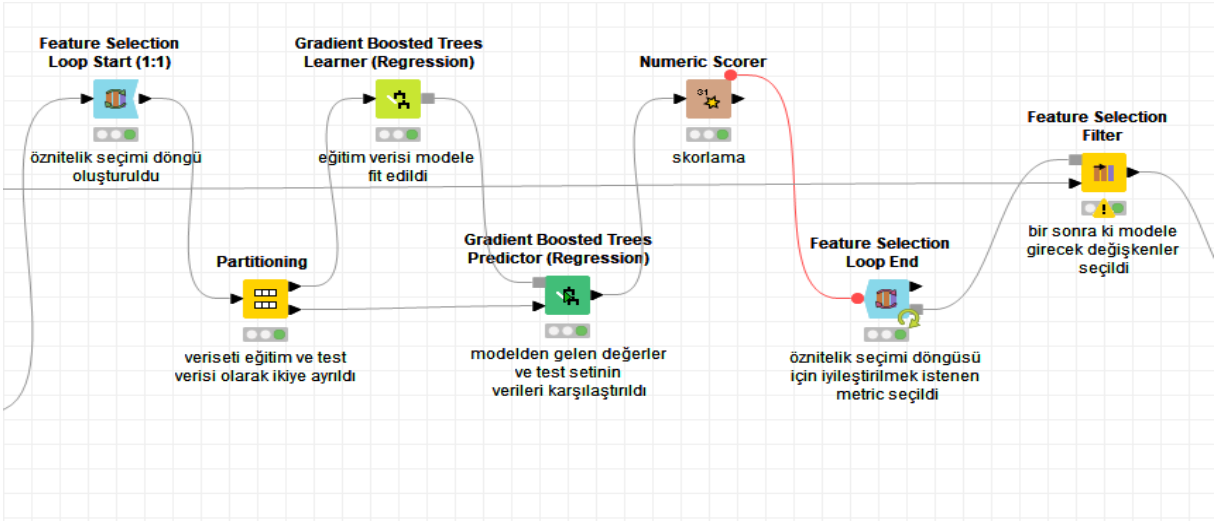
Şekil 23 – Skorlama 2

Polinominal Regresyon algoritmasının son skoru şekil 23'te gibidir. Görüldüğü üzere 0,678'den 0,819 oranına kadar artmış bir R^2 skoru gözükmektedir. Yaptığımız uygulamaların işe yaradığını var sayabiliriz.

3.2.8 Gradyan Destekli Ağaç Regresyonu

Gradyan Destekli Ağaç Regresyonu projemizde kullandığımız 3 makine öğrenmesinden ikincisidir. Önce Öznitelik Seçimi sonra ise Parametre optimizasyonu yaparak bu makine öğrenmesi algoritmasında verim elde etmeye çalışacağız.

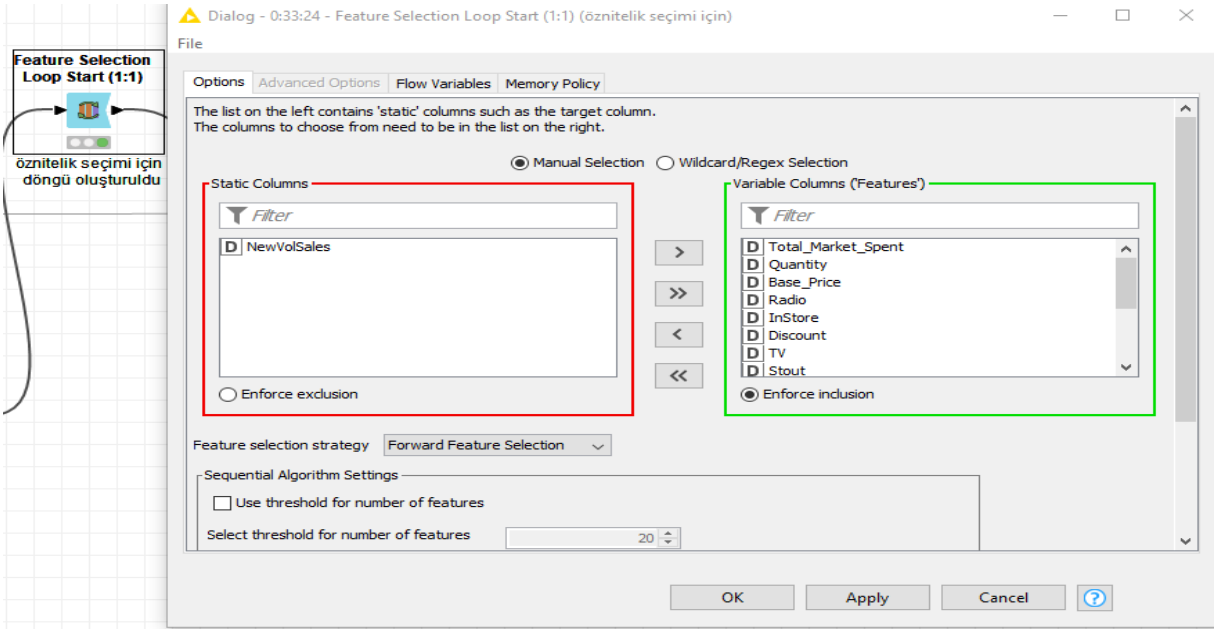
3.2.8.1 Gradient Boost Regression Feature Selection MetaNod'u



Şekil 24 – Öznitelik Seçimi 2

Feature Selection MetaNod'u şekil 24'de görüldüğü gibidir.

a) Öznitelik Seçimi Döngüsü Başlangıcı

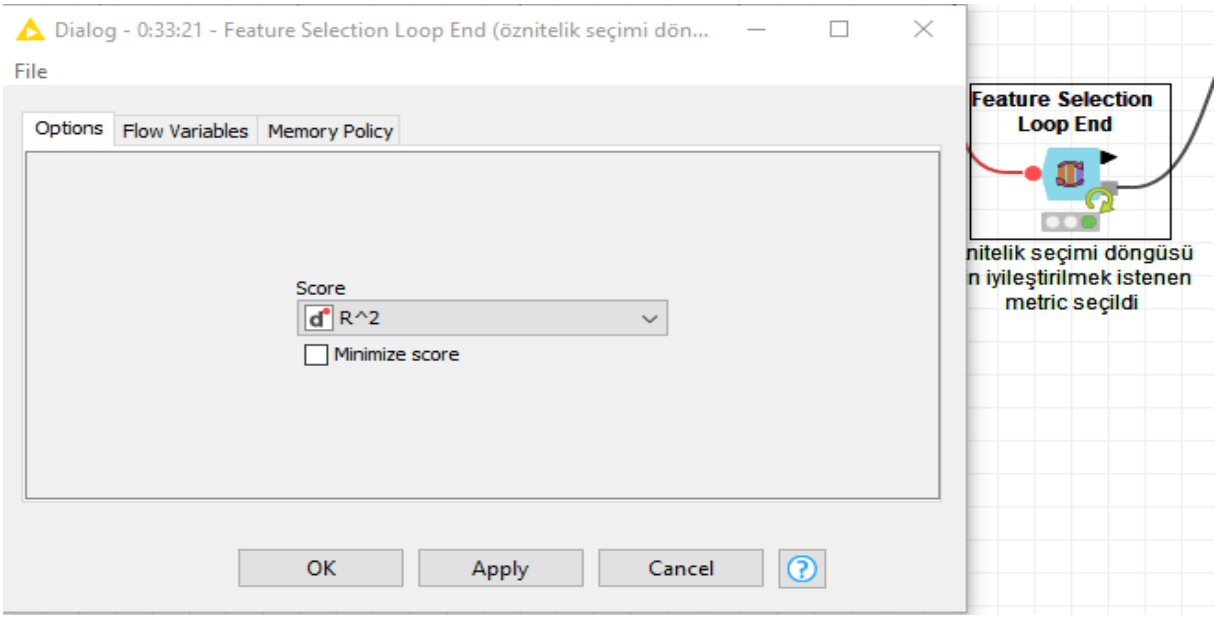


Şekil 25– Öznitelik Seçim Döngüsü Başlangıcı 2

Feature Selection döngüsünü çalıştırmak için hedef değişkenimizi Static Columns sekmesine atıyoruz ve Feature Selection Strategy seçeneklerinden Forward Feature Selection'ı seçiyoruz.

a) Öznitelik Seçimi Döngüsü Bitişi

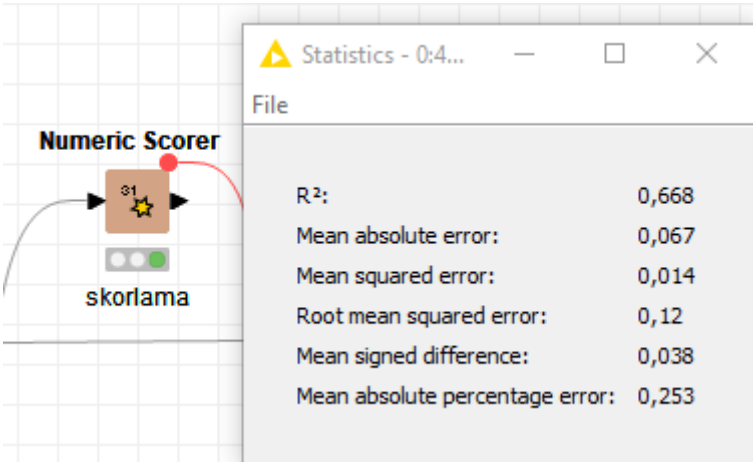
Makine öğrenmesi algoritmasını çalıştırmadan önce döngünün biteceği Nod'da maksimize veya minimize etmek istediğimiz metriği seçmemiz gerekmektedir.



Şekil 26 – Öznitelik Seçim Döngüsü Başlangıcı 2

Öznitelik seçimi döngüsünün sonunda R^2 metriğinin maksimize edilmesini tercih ettik bizim başarı metriğimiz R^2 metriği olacaktır.

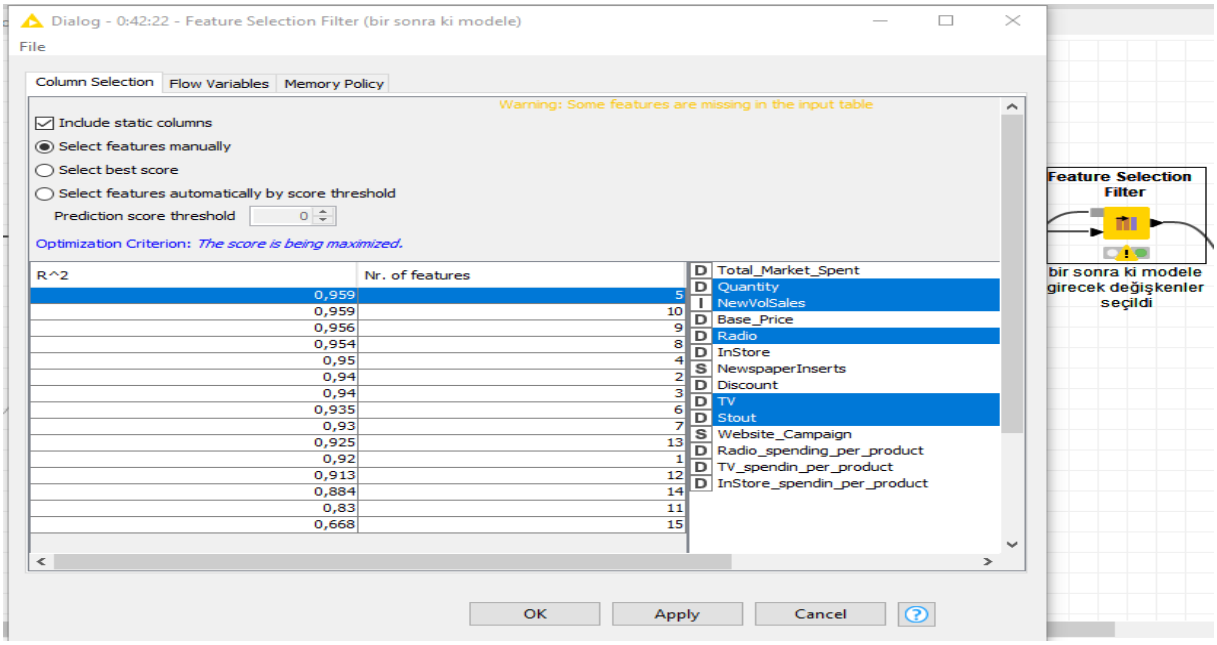
a) Makine Öğrenmesi Performansının Skorlanması



Şekil 27- Skorlama 3

Gradyan Destekli Ağaç Regresyonu algoritmasından aldığımız ilk sonuçlar bu şekildedir. Elde Edilen sonuçların iyileştirilmeye ihtiyaç duyduğunu görmekteyiz.

a) Öznitelik Seçimi Döngüsü Sonrası Değişken Seçimi

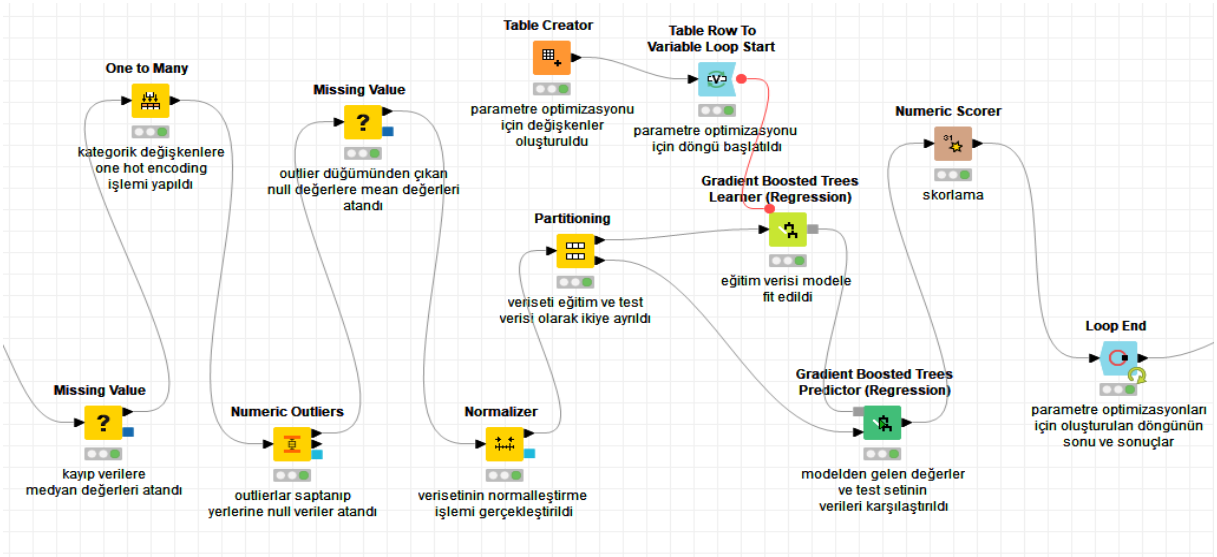


Şekil 28 – Öznitelik Filtresi 2

Şekil 28'de de görüldüğü üzere Feature Selection Filter Nod'unda değişkenlerin R^2 metrik değerleri, bu değerlerin ortaya çıkmasını sağlayan değişkenler ve kaç değişken kullanıldığı gösterilmektedir. Bu değişkenler arasından istediğimiz değişken kombinasyonlarından birini seçip yeni kuracağımız modele yönlendirebiliriz.

3.2.8.2 Gradient Boost Regression Parameter Optimization Metanod'u

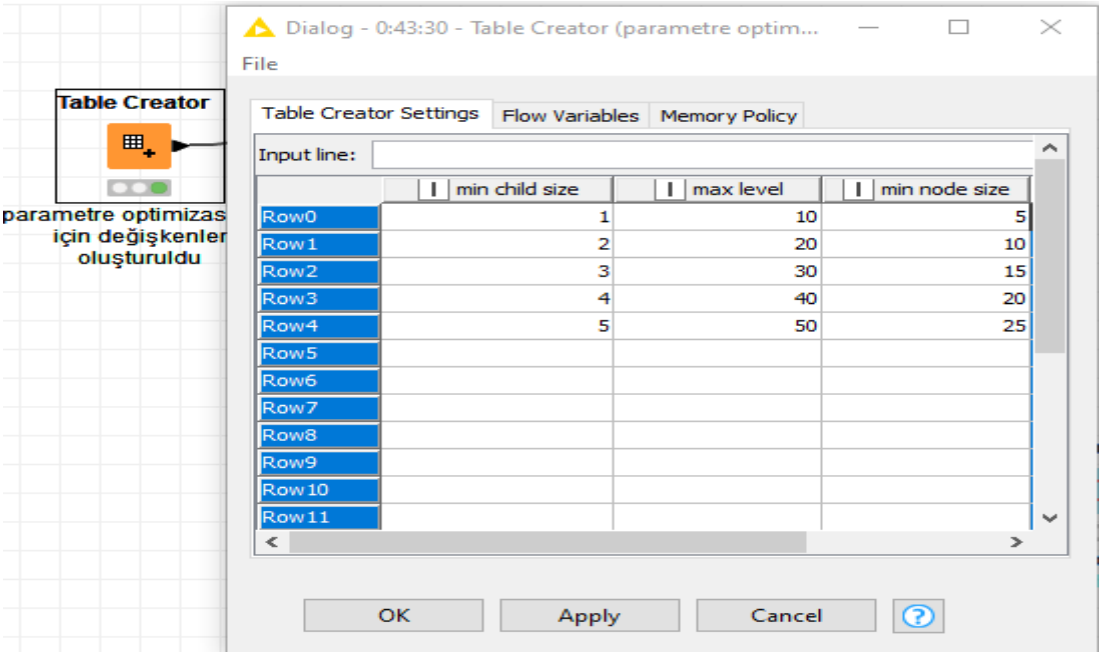
Parametre optimizasyonu, algoritmamız eğitim işlemi gerçekleştirirken kullandığı hazır parametreler yerine dışardan manuel olarak farklı parametreler girilmesine denir. Bu işlem sonucunda algoritmamızın başarısının artması beklenmektedir.



Şekil 29 – Parametre Optimizasyonu 2

Şekil 29'da görüldüğü üzere bu MetaNod'da bulunan ilk 5 nod daha önce yaptığımız veri ön hazırlık işlemleridir.

a) Parametre Optimizasyonu İçin Değişken Oluşturma

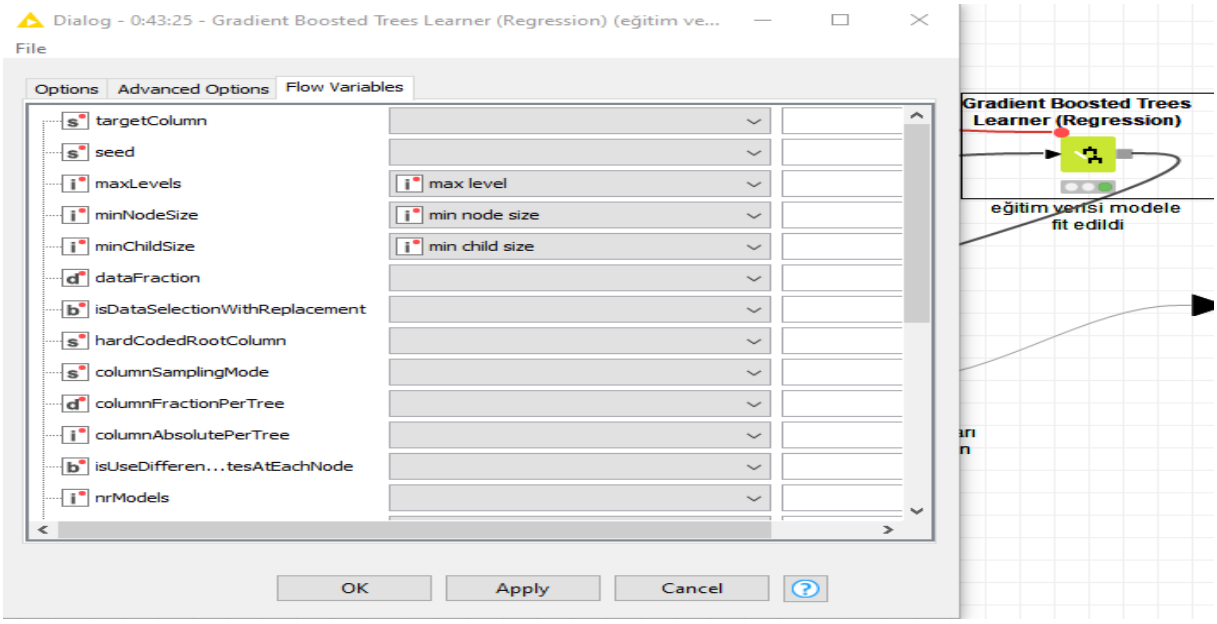


Şekil 30 – Parametre Oluşturma 2

Algoritmamızın içine yerleştirmek istediğimiz yeni parametrelerimiz hazırdır. Döngü bu değerleri teker teker deneyip sonuçlarını bize getirecektir.

b) Parametrenin Algoritmaya Yerleştirilmesi

Makine öğrenmesi algoritmamızın parametrelerini optimize edebilmek için oluşturduğumuz değerleri modelimize eklememiz gerekmektedir.

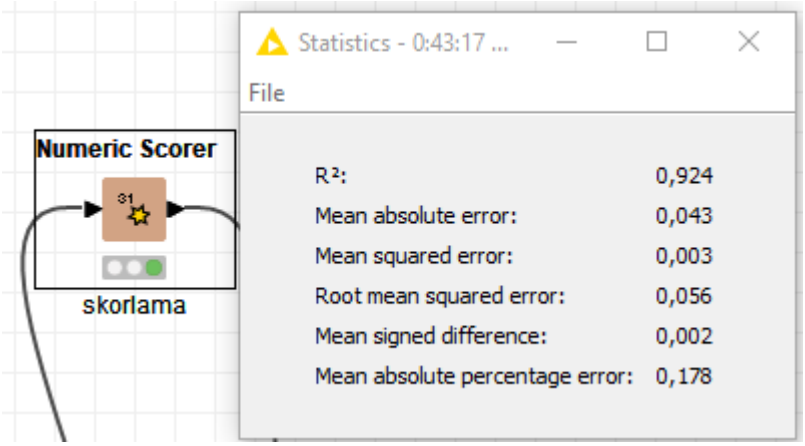


Şekil 31 – Parametre Yerleştirme 2

Gradient Boost Trees Learner Nod'unda bulunan Flow Variables sekmesindeki hazır olarak gelen değişkenlerle daha önceden hazırladığımız değişkenleri değiştiriyoruz.

c) Makine Öğrenmesi Performansının Skorlanması

Yaptığımız parametre optimizasyonu ve öznitelik seçimi uygulamaların sonucunu göreceğimiz son skorlama nod'u olacaktır.



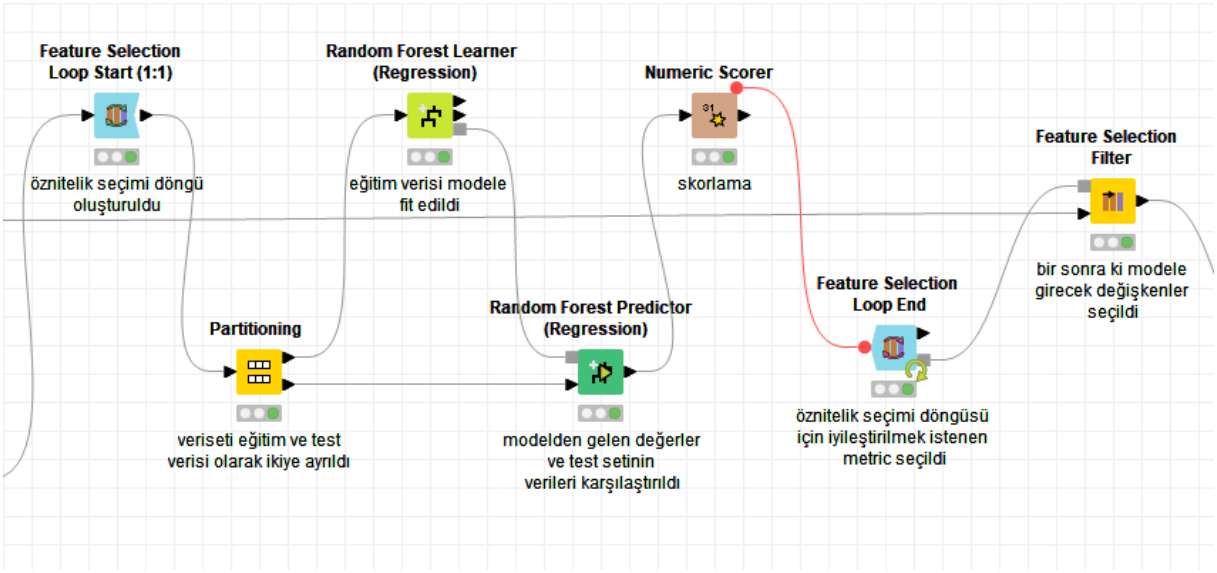
Şekil 32 – Skorlama 4

Gradyan Destekli Ağaç Regresyonu algoritmasının son skoru şekil 32'deki gibidir. Görüldüğü üzere 0,668'den 0,924 oranına kadar artmış bir R^2 skoru gözükmektedir. Yaptığımız uygulamaların işe yaradığını var sayabiliriz.

3.2.9 Rassal Orman Regresyonu

Rassal Orman Regresyonu projemizde kullandığımız 3 makine öğrenmesinden sonuncusudur. Önce Öznitelik Seçimi sonra ise Parametre optimizasyonu yaparak bu makine öğrenmesi algoritmasında verim elde etmeye çalışacağız.

3.2.9.1 Random Forest Regression Feature Selection Metanod'u

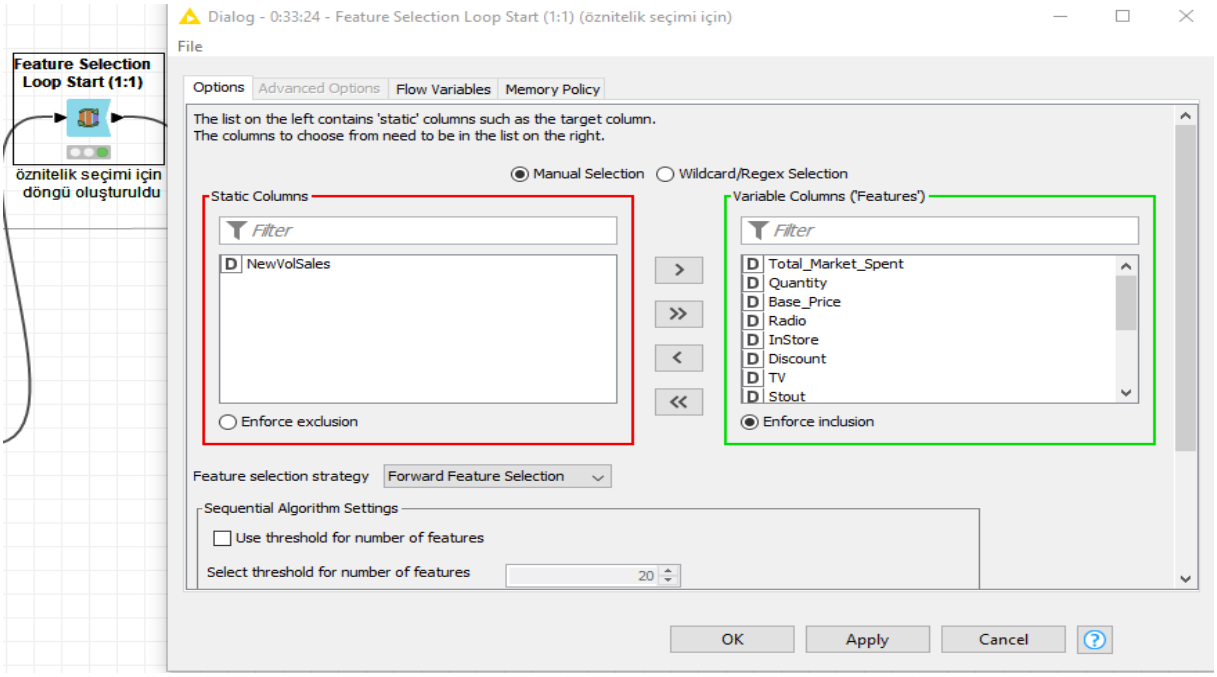


Şekil 33 – Öznitelik Seçimi 3

Feature Selection MetaNod'u şekil 33'te görüldüğü gibidir.

a) Öznitelik Seçimi Döngüsü Başlangıcı

Feature Selection Loop Start Nod'uyla öznitelik seçimi döngümüzü başlatıyoruz.

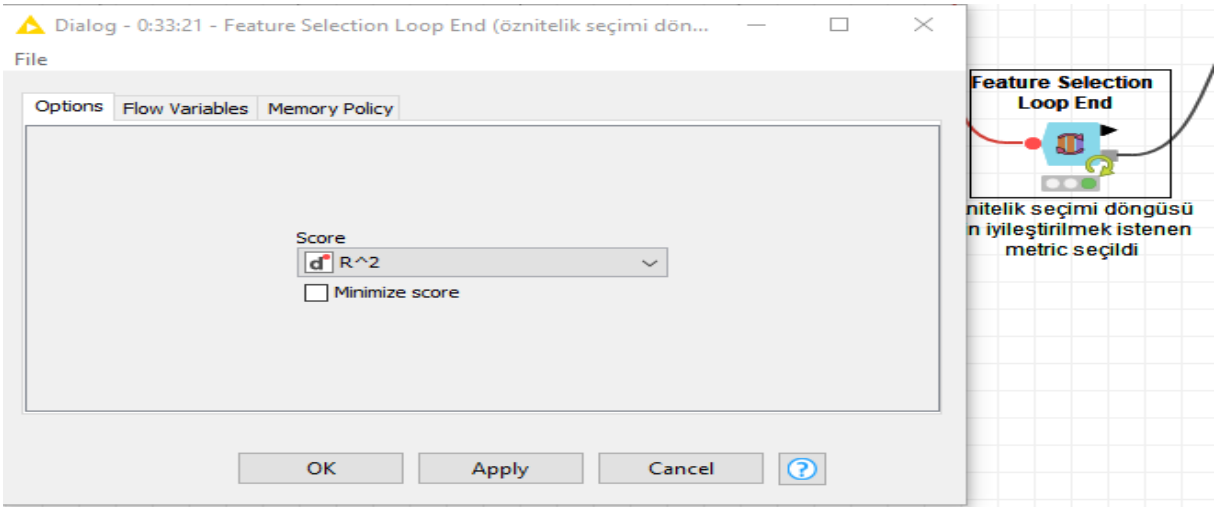


Şekil 34 – Öznitelik Seçim Döngüsü Başlangıcı 3

Feature Selection döngüsünü çalıştırmak için hedef değişkenimizi Static Columns sekmesine atıyoruz ve Feature Selection Strategy seçeneklerinden Forward Feature Selection'ı seçiyoruz.

a) Öznitelik Seçimi Döngüsü Bitişi

Makine öğrenmesi algoritmasını çalıştırmadan önce döngünün biteceği Nod'da maksimize veya minimize etmek istediğimiz metriği seçmemiz gerekmektedir.

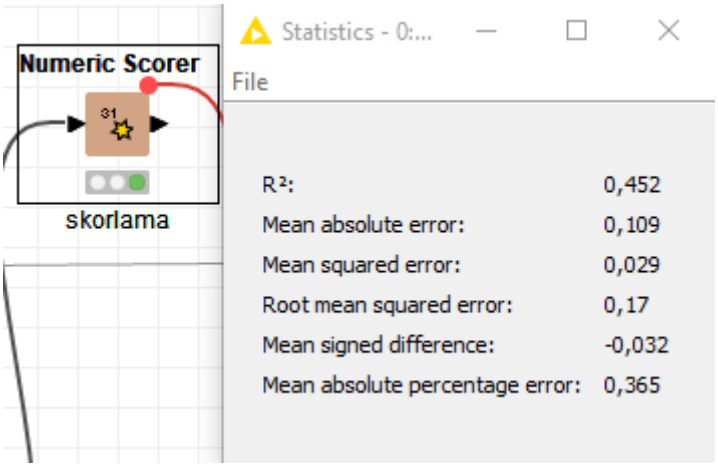


Şekil 35 – Öznitelik Seçim Döngüsü Başlangıcı 3

Öznitelik seçimi döngüsünün sonunda 'R^2' metriğinin maksimize edilmesini tercih ettik bizim başarı metriğimiz 'R^2' metriği olacaktır.

a) Makine Öğrenmesi Performansının Skorlanması

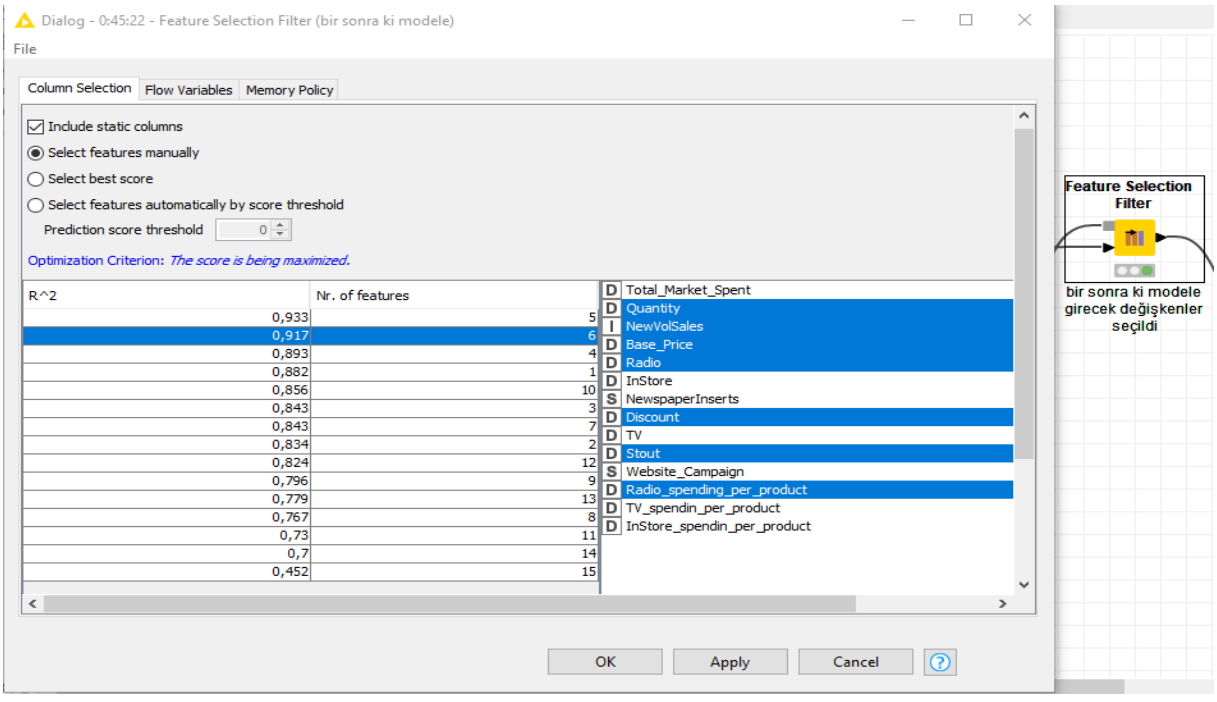
Numeric Scorer Nod'unda eğitimden çıkmış verilerle test verilerini karşılaştırıp sonuçlara bakabildiğimiz Nod'dur.



Şekil 36 - Skorlama 5

Rassal Orman Regresyonu algoritmasından aldığımız ilk sonuçlar bu şekildedir. Elde Edilen sonuçların iyileştirilmeye ihtiyaç duyduğunu görmekteyiz ve bu algoritmanın şuana kadar en düşük başarı skorunu verdiğini söyleyebiliriz.

a) Öznitelik Seçimi Döngüsü Sonrası Değişken Seçimi

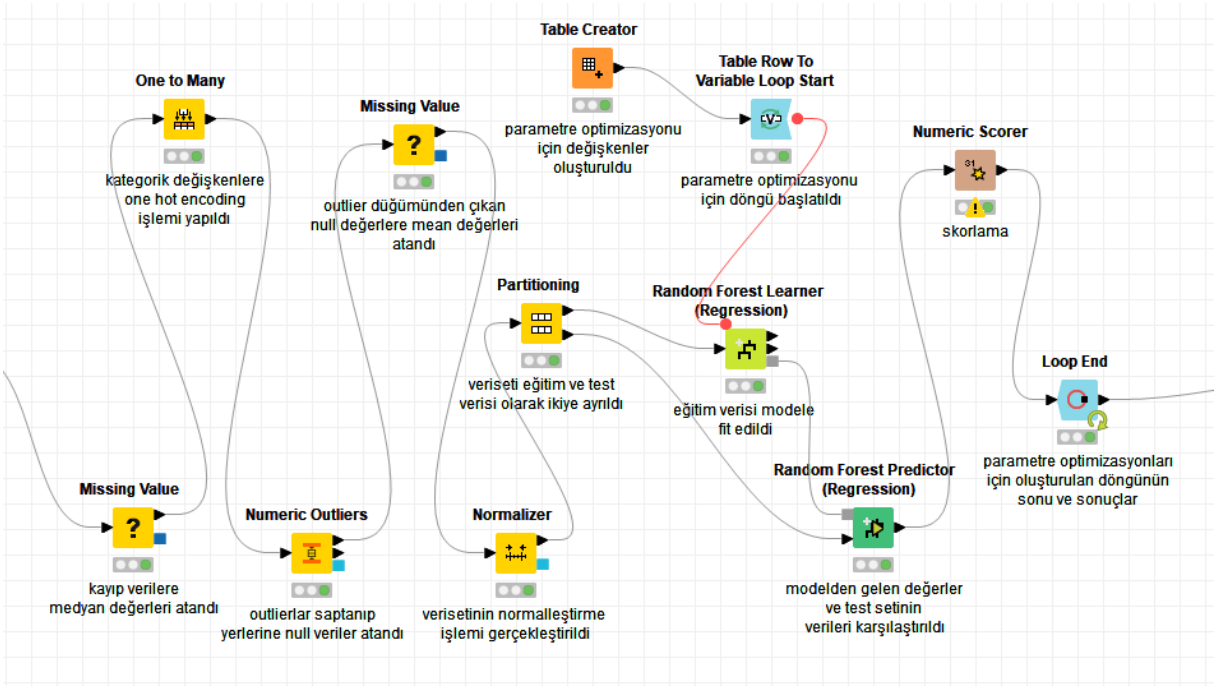


Şekil 37 – Öznitelik Seçimi 3

Şekil 36'da da görüldüğü üzere Feature Selection Filter Nod'unda değişkenlerin R^2 metrik değerleri, bu değerlerin ortaya çıkmasını sağlayan değişkenler ve kaç değişken kullanıldığı gösterilmektedir. Bu değişkenler arasından istediğimiz değişken kombinasyonlarından birini seçip yeni kuracağımız modele yönlendirebiliriz.

3.2.9.2 Random Forest Regression Parameter Optimization Metanod'u

Parametre optimizasyonu, algoritmamız eğitim işlemi gerçekleştirirken kullandığı hazır parametreler yerine dışardan manuel olarak farklı parametreler girilmesine denir. Bu işlem sonucunda algoritmamızın başarısının artması beklenmektedir.

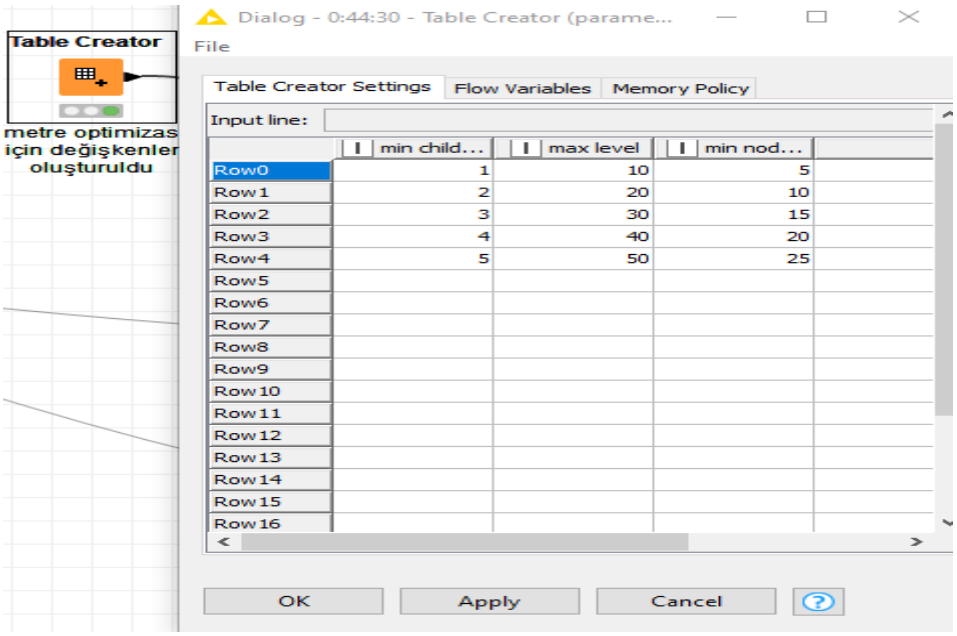


Şekil 38 – Parametre Optimizasyonu 3

Şekil 38'de görüldüğü üzere bu MetaNod'da bulunan ilk 5 nod daha önce yaptığımız veri ön hazırlık işlemleridir.

a) Parametre Optimizasyonu İçin Değişken Oluşturma

Daha öncede kullanmış olduğumuz bu nod Knime içerisinde manuel olarak tablolar oluşturmaya imkan sağlıyor.

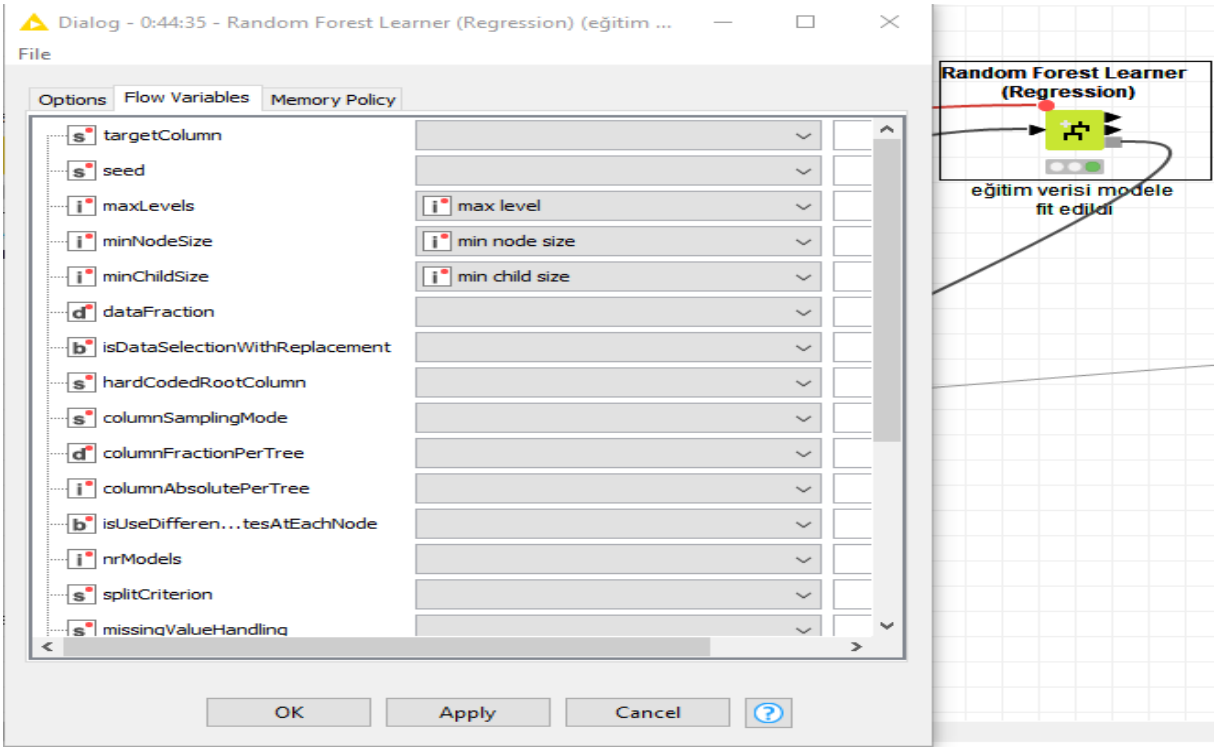


Şekil 39 – Parametre Oluşturma 3

Algoritmamızın içine yerleştirmek istediğimiz yeni parametrelerimiz hazır. Döngü bu değerleri teker teker deneyip sonuçlarını bize getirecektir.

a) Parametrenin Algoritmaya Yerleştirilmesi

Makine öğrenmesi algoritmamızın parametrelerini optimize edebilmek için oluşturduğumuz değerleri modelimize eklememiz gerekmektedir.

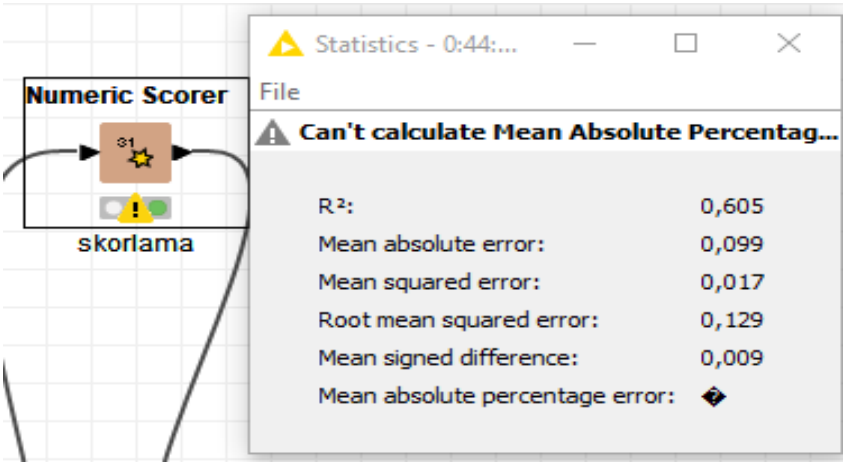


Şekil 40- Parametre Yerleştirme 3

Random Forest Learner Nod'unda bulunan Flow Variables sekmesindeki hazır değişkenlerle daha önceden hazırladığımız değişkenlerle değiştiriyoruz.

a) Makine Öğrenmesi Performansının Skorlanması

Yaptığımız parametre optimizasyonu ve öznitelik seçimi uygulamaların sonucunu göreceğimiz son skorlama nod'u olacaktır.



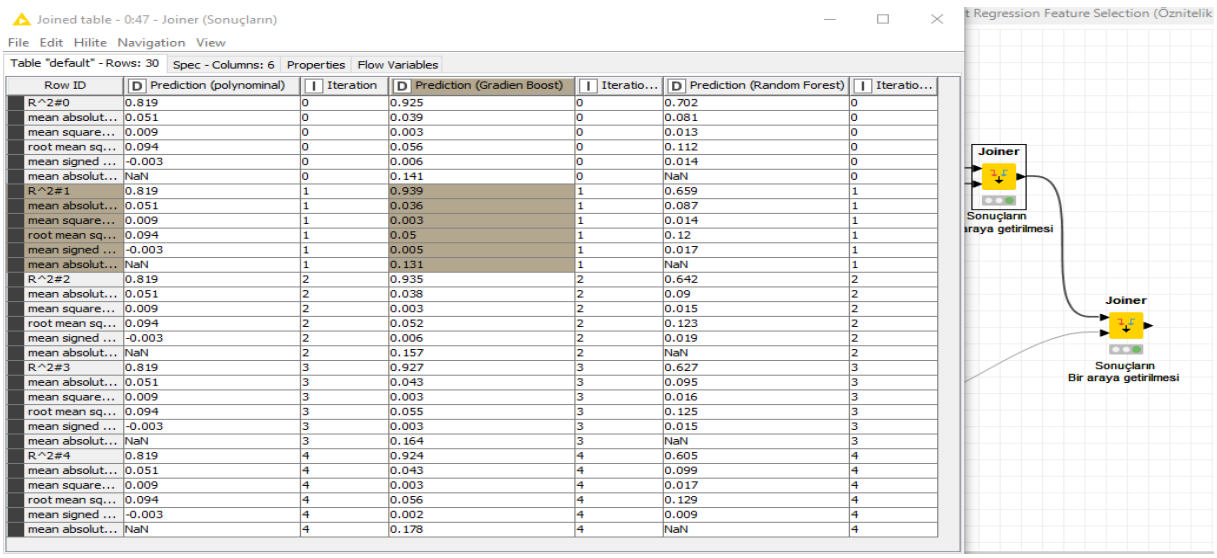
Şekil 41 – Skorlama 6

Rassal Orman Regresyonu algoritmasının son skoru şekil 41'deki gibidir. Görüldüğü üzere 0,452'den 0,605 oranına kadar artmış bir R^2 skoru gözükmektedir. Yaptığımız uygulamaların işe yaradığını var sayabiliriz. Ama bu artışın yeterli olduğunu söylememiz çok güçtür.

3.2.10 Elde Edilen Bütün Sonuçların Karşılaştırılması

Kurduğumuz 3 ayrı modelden farklı farklı sonuçlar almış bulunmaktayız. Bu sonuçları karşılaştırmak için bir araya getirmemiz gerekmektedir.

3.2.10.1 Elde Edilen Bütün Sonuçların Birleştirilmesi



Row ID	Prediction (polynominal)	Iteration	Prediction (Gradien Boost)	Iteratio...	Prediction (Random Forest)	Iteratio...
R^2#0	0.819	0	0.925	0	0.702	0
mean absolut...	0.051	0	0.039	0	0.081	0
mean square...	0.009	0	0.003	0	0.013	0
root mean sq...	0.094	0	0.056	0	0.112	0
mean signed ...	-0.003	0	0.006	0	0.014	0
mean absolut...	NaN	0	0.141	0	NaN	0
R^2#1	0.819	1	0.939	1	0.659	1
mean absolut...	0.051	1	0.036	1	0.087	1
mean square...	0.009	1	0.003	1	0.014	1
root mean sq...	0.094	1	0.05	1	0.12	1
mean signed ...	-0.003	1	0.005	1	0.017	1
mean absolut...	NaN	1	0.131	1	NaN	1
R^2#2	0.819	2	0.935	2	0.642	2
mean absolut...	0.051	2	0.038	2	0.09	2
mean square...	0.009	2	0.003	2	0.015	2
root mean sq...	0.094	2	0.052	2	0.123	2
mean signed ...	-0.003	2	0.006	2	0.019	2
mean absolut...	NaN	2	0.157	2	NaN	2
R^2#3	0.819	3	0.927	3	0.627	3
mean absolut...	0.051	3	0.043	3	0.095	3
mean square...	0.009	3	0.003	3	0.016	3
root mean sq...	0.094	3	0.055	3	0.125	3
mean signed ...	-0.003	3	0.003	3	0.015	3
mean absolut...	NaN	3	0.164	3	NaN	3
R^2#4	0.819	4	0.924	4	0.605	4
mean absolut...	0.051	4	0.043	4	0.099	4
mean square...	0.009	4	0.003	4	0.017	4
root mean sq...	0.094	4	0.056	4	0.129	4
mean signed ...	-0.003	4	0.002	4	0.009	4
mean absolut...	NaN	4	0.178	4	NaN	4

Şekil 42 – Sonuçların Birleştirilmesi

Modellerimizin Parametre Optimizasyon Metonod'larını joinerlarla birleştirerek şekil 42'deki değerleri elde ettik. En iyi metrik değerlerini Gradyan Destekli Ağaç Regresyonu Algoritmasının 2. İterasyonunda elde ettiğimiz gözüküyor. Polinomnial Regresyon algoritmasında iterasyon sayısındaki değişim sonuçlara etki etmediğini gözlemleniyor. Rassal Orman Regresyon Algortimasında ise iterasyon değerleri arttıkça sonuçların azaldığını söylenebilir.

3.3 Python ile Uçtan Uca Veri Analizi ve Esneklik Değerlerinin Hesaplanması

3.3.1 Kütüphanelerin Yüklenmesi

Python programlama diliyle yapacağımız işlemleri kütüphaneler aracılığıyla yapacağız. Bunun için bu kütüphaneleri projemize yüklememiz etmemiz gerekmektedir.

```
#İmporting libaries
```

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.formula.api as sm
import xgboost as xgb

from sklearn.metrics import mean_squared_error,r2_score, mean_absolute_error
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, MinMaxScaler

from sklearn.ensemble import AdaBoostRegressor
from xgboost.sklearn import XGBRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn import neighbors
from sklearn.neural_network import MLPRegressor
```

```
import warnings
warnings.filterwarnings("ignore")
```

Kod 1 – Kütüphanelerin yüklenmesi

Kod 1'de de görüldüğü üzere veri ön hazırlık işlemlerinden değerlendirme işlemlerine kadar kullanmamız gereken bütün kütüphaneleri projemize yükledik.

3.3.2 Verimizi okutmak

Verimizi projemizin içinde okutmak için "pandas" kütüphanesinin "read" methodunu kullanacağız.

```
data = pd.read_csv("mktmix.csv")
```

```
data.head()
```

	NewVolSales	Base_Price	Radio	InStore	NewspaperInserts	Discount	TV	Stout	Website_Campaign
0	19564	15.029276	245.0	15.452	NaN	0.000	101.780000	2.28342	NaN
1	19387	15.029276	314.0	16.388	NaN	0.000	76.734000	2.22134	NaN
2	23889	14.585093	324.0	62.692	NaN	0.050	131.590200	2.00604	NaN
3	20055	15.332887	298.0	16.573	NaN	0.000	119.627060	2.19897	NaN
4	20064	15.642632	279.0	41.504	NaN	0.045	103.438118	1.81860	NaN

Kod 2 – Verinin okunması

Pandas kütüphanesinin "read" methodunu kullanarak verimizi projemize okuttuk ve "head" methoduyla görünümünü gözden geçirdik.

3.3.3 Veri Hakkında Bilgi Edinmek

Veri hakkında genel bilgi edinmek için birkaç sorgulama işlemi yapacağız ve veri ön hazırlığı için hangi işlemleri yapmamız gerektiğine karar vereceğiz.

a) Veri tiplerinin öğrenilmesi

Veri kümemizdeki veri tiplerini öğrenmek için "info" komutunu kullanacağız.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 104 entries, 0 to 103
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   NewVolSales        104 non-null    int64  
 1   Base_Price         104 non-null    float64
 2   Radio              100 non-null    float64
 3   InStore            104 non-null    float64
 4   NewspaperInserts   6 non-null      object 
 5   Discount           104 non-null    float64
 6   TV                 104 non-null    float64
 7   Stout              104 non-null    float64
 8   Web                14 non-null     object 
dtypes: float64(6), int64(1), object(2)
memory usage: 7.4+ KB
```

```
data.shape
```

```
(104, 9)
```

Kod 3 – Değişken Tipleri

Kod 3'de de görüldüğü üzere verimizde bulunan değişkenlerden 6'sı Float, 1'i İnteger ve 2 değişkenin de Object tipinde olduğu görülmüştür. Float değişkenlerin numerik değişkenler, Object değişkenlerin kategorik değişkenler ve Integer değişkenin ise bizim tahmin etmeyi hedeflediğimiz değişken olduğunu söyleyebiliriz.

Aynı zamanda "shape" komutuyla verimizde kaç adet gözlem ve kaç adet değişken olduğunu görebiliyoruz. 104 adet gözlemimiz ve 9 adet değişkenimiz olduğu gözlemlenmiştir.

b) Kayıp Verilerin Gözlenmesi ve Düzeltilmesi

Veri kümelerinde bulunan kayıp veriler veri ön hazırlık aşamalarında sorunlar çıkarmaktadır. Bu sorunların üstesinden gelmek için veri kümemizde olabilecek kayıp verilerin kontrolü ve bu kayıp verilerin düzenlenmesi işlemlerini yapmamız gerekmektedir.

```
data.isnull().sum()
```

```
NewVolSales          0
Base_Price           0
Radio                4
InStore              0
NewspaperInserts    98
Discount             0
TV                   0
Stout                0
Web                 90
dtype: int64
```

```
data["Web"] = data["Web"].fillna("other")
data["NewspaperInserts"] = data["NewspaperInserts"].fillna("other")
data["Radio"] = data["Radio"].fillna(data["Radio"].median())
```

Kod 4 – Kayıp Veri Kontrolü

Kod 4'de de görüldüğü üzere "Radio" değişkeninde 4 adet, "NewspaperInserts" değişkeninde 98 adet ve "Web" değişkeninde 90 adet kayıp veri bulunmaktadır. Bu değişkenlerdeki kayıp verilerin düzetilmesi işlemi "fillna" metoduyla yapılmıştır. "Radio" değişkenin medyan değerleri ve diğer iki değişkenlerdeki kayıp verileri de "other" değeri atanmıştır.

c) Numerik Değerlerin İstatistiklerinin Gözlemlenmesi

Veri kümemizde bulunan istatistiksel değerlere bakarak veri hakkında birçok değeri gözlemleyebiliriz. Bunun için "describe" metodunu kullanacağız.

```
data.describe()
```

	NewVolSales	Base_Price	Radio	InStore	Discount	TV	Stout
count	104.000000	104.000000	104.000000	104.000000	104.000000	104.000000	104.000000
mean	20171.067308	15.306740	257.528846	32.918567	0.022059	141.009774	2.545966
std	1578.604670	0.528902	85.392805	13.682570	0.027668	42.949231	0.310070
min	17431.000000	13.735724	0.000000	10.782000	0.000000	37.656174	1.818600
25%	19048.750000	15.029276	235.750000	22.183750	0.000000	117.108343	2.316450
50%	19943.500000	15.332887	278.500000	31.161500	0.000000	138.581542	2.502120
75%	20942.750000	15.642632	312.250000	41.079000	0.049423	175.681780	2.814315
max	24944.000000	16.281020	399.000000	68.119000	0.090763	240.291967	3.158620

Kod 5 – İstatistikler

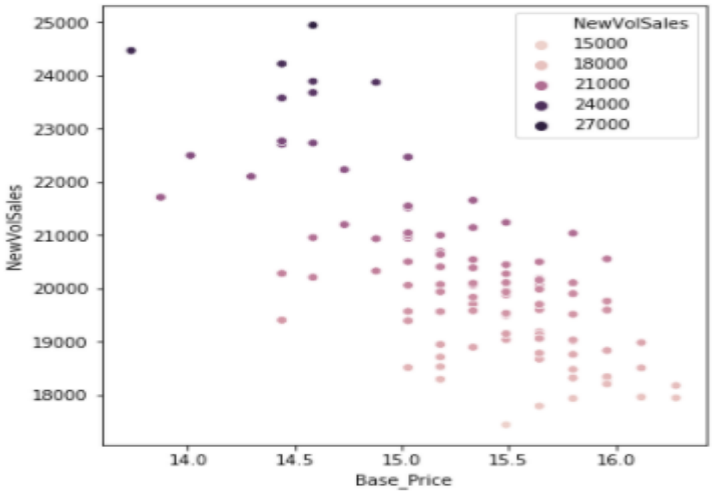
Kod 5'de veri kümemizdeki numerik değişkenlerin birçok istatistiksel değerlerini görmekteyiz. Veri kümemizde standartlaştırma işlemini yapmamız gerektiğini anlayabiliyoruz. Bu kanıya varılmasının sebebi "Radio" değişkenin değerleri ile "Discount" değişkenlerinin arasında bulunan sayısal farklılıklardır. Bu sayısal farklılıklar düzenlenmezse makine öğrenmesi sürecinde düşük değerlerin etkileri de düşük olacaktır. Bu durum algoritmamızı yanlış yönlendireceği için bu durumu standartlaştırma işlemi yaparak düzenlememiz gerekmektedir.

3.3.4 Görselleştirme

Veri kümemiz üzerinde görselleştirmeler yaparak kullanacağımız veri kümesi hakkında daha çok bilgi sahibi olabiliriz.

a) Dağılım Grafiği 1

```
plt.figure(figsize=(6,6))
sns.scatterplot(x=data['Base_Price'],y=data['NewVolSales'],hue=data['NewVolSales'])
plt.xlabel('Base_Price')
plt.ylabel('NewVolSales')
plt.show()
```

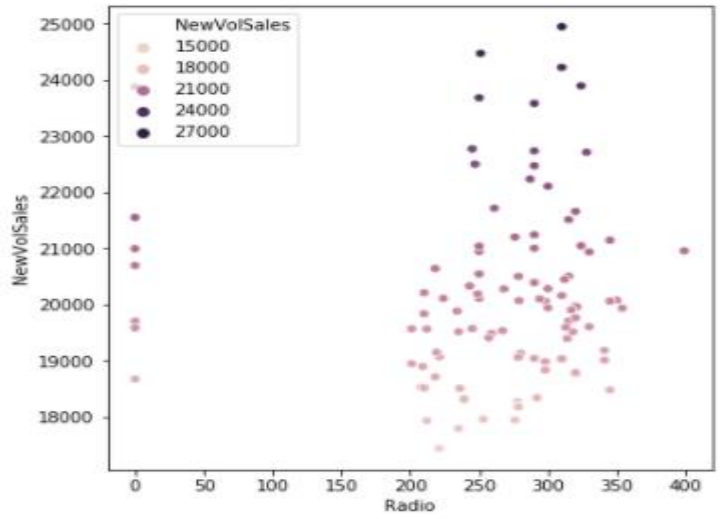


Kod 6 – Görsel 1

Kod 6'da x eksenimizde hedef değişkenimiz olan NewVolSales değişkeni, y ekseninde ise Haftalık birim fiyatı belirten Base_Price değişkenimizdir. Bu durumda bu şekilden haftalık birim fiyatın düşük olduğu haftalarda toplam satışın yüksek olduğunu, haftalık birim satış fiyatının yüksek olduğu haftalarda toplam satışın az olduğunu gözlemleyebiliriz.

b) Dağılım Grafiği 2

```
plt.figure(figsize=(6,6))
sns.scatterplot(x=data['Radio'],y=data['NewVolSales'],hue=data['NewVolSales'])
plt.xlabel('Radio')
plt.ylabel('NewVolSales')
plt.show()
```

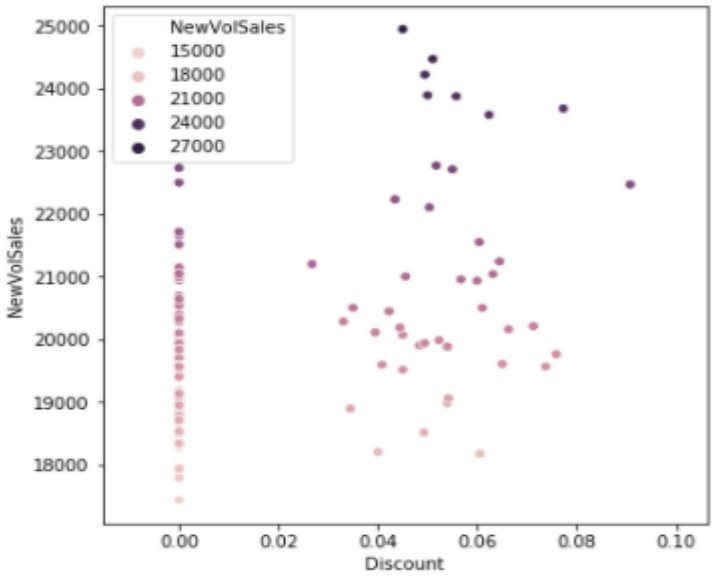


Kod 7 – Görsel 2

Kod 7'deki görsele bakınca "Radio" değişkenin de bazı gözlemlerin 0 değeri içerdiğini gözlemleyebiliriz.

c) Dağılım Grafiği 3

```
plt.figure(figsize=(6,6))
sns.scatterplot(x=data['Discount'],y=data['NewVolSales'],hue=data['NewVolSales'])
plt.xlabel('Discount')
plt.ylabel('NewVolSales')
plt.show()
```



Kod 8 – Görsel 3

Kod 8' deki görsele bakınca "Discount" gözlemlerinden bazılarının 0 değeri içerdiği gözlemlenmiştir.

d) Korelasyon Matrisi

```
corr_matrix = data.corr()
plt.figure(figsize=(5,5))
sns.clustermap(corr_matrix, annot = True, fmt = ".2f")
plt.title("Correlation between columns")
plt.show
```

Kod 9 – Görsel 4

Kod 9'daki görselde de görüldüğü "Discount" ve "InStore" değişkenleri arasında ilişki 0.72'lik pozitif korelasyon, "NewVolSales" ile "Base_Price" değişkenleri arasında da 0.72'lik negatif ilişki gözlemlenmektedir.

3.3.5 Veri Ön Hazırlık

Modelimizi kurmaya geçmeden önce bir veri ön hazırlık işlemlerini yapmamız gerekmektedir.

a) Kategorik Değişkenlerin Numerik Değişkenlere Dönüştürülmesi (One – Hot Encoding)

Veri kümemizde 2 adet kategorik değişken bulunmaktaydı bu değerleri modelimizi kurarken olumsuz etkilememesi için numerik değerlere çevireceğiz.

```
# Editing categorical variables
```

```
cat_col = ["Web","NewspaperInserts"]
data = pd.get_dummies(data, columns = cat_col)
```

```
data = data.drop(["Web_other","NewspaperInserts_other"], axis = 1)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 104 entries, 0 to 103
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   NewVolSales               104 non-null    int64
 1   Base_Price                104 non-null    float64
 2   Radio                     104 non-null    float64
 3   InStore                   104 non-null    float64
 4   Discount                  104 non-null    float64
 5   TV                        104 non-null    float64
 6   Stout                     104 non-null    float64
 7   Web_Facebook              104 non-null    uint8
 8   Web_Twitter               104 non-null    uint8
 9   Web_Website Campaign      104 non-null    uint8
 10  NewspaperInserts_Insert   104 non-null    uint8
dtypes: float64(6), int64(1), uint8(4)
memory usage: 6.2 KB
```

Kod 10 – Kategorik Değişkenlerin Düzenlenmesi

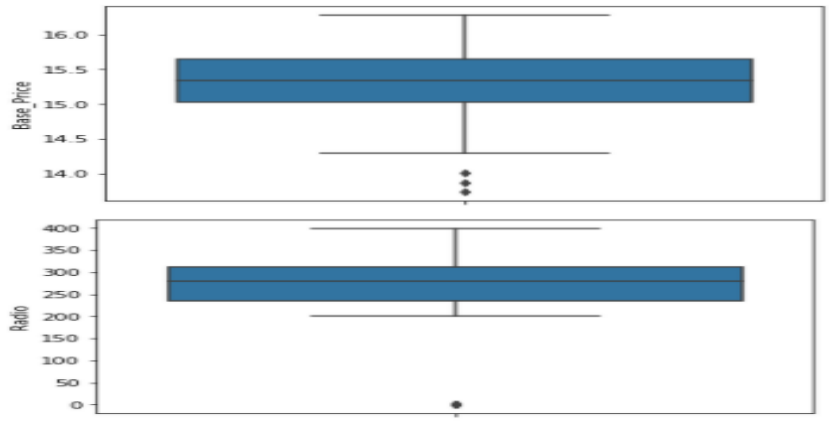
Kod 10'da da görüldüğü üzere ilk önce kategorik değişkenlerimizi bir değere atadık. Bu değere atama işlemini yaptıktan sonra "pandas" kütüphanesinin "get_dummies" methoduyla bu değişkenler içinde bulunan kategorik değişkenleri numerik değişkenlere çevirdik. Bu çevirme işleminin sonunda kukla değişken tuzağına düşmemek için her iki değişken grubundan birer değişkeni veri kümesinden düşürdük.

b) Aykırı Değerlerin Düzenlenmesi

```
# Handling outliers
```

```
chechk_outliers = ["Base_Price","Radio","InStore","Discount","TV","Stout"]
```

```
for c in chechk_outliers:
    plt.figure()
    sns.boxplot(x = c, data = data, orient = "v")
```



Kod 11 – Aykırı Değer Tespiti

Veri kümemizdeki aykırı değerleri görebilmek için "box plot" grafiğinden yararlandık. İlk önce verimizde bulunan numerik değişkenleri bir değere atadık bu değerleri "for" döngüsüyle birer birer "box plot" grafiklerini oluşturduk. Bu grafiklerde görüldüğü üzere "Radio" ve "Base_Price" değişkenlerinde aykırı değerler gözlemlenmiştir.

```
outliers = ["Radio","Base_Price"]
```

```
def treat_outlier(x):
    q5 = np.percentile(x,5)
    q25= np.percentile(x,25)
    q75= np.percentile(x,75)
    up_trend=np.percentile(x,95)
    IQR = q75-q25
    low_level = q25-(1.5*IQR)
    up_level = q75+(1.5*IQR)

    return x.apply(lambda y: up_trend if y> up_level else y).apply(lambda y: q5 if y < low_level else y)
```

```
for i in data[outliers]:
    data[i]=treat_outlier(data[i])
```

Kod 12 – Aykırı Değerlerin Düzeltilmesi

Kod 12'de de görüldüğü üzere "box plot" grafiklerinden elde ettiğimiz bilgiyle aykırı değer içeren değişkenleri bir değere atadık. Bu işlemden sonra bir fonksiyon tanımlıyoruz. Bu fonksiyon sayesinde verimizin normal dağılım eğrisindeki yüzde 5'lik, 25'lik, 75'lik ve 95'lik değerlerini "numpy" kütüphanesinin "percentile" metoduyla belirlemiş olduk. Eğer aykırı değerimiz %5lik değerler içerisinde ise bu değeri "low_level" değeri ile eşitlemesini istedik. Eğer aykırı değerimiz %95'lik değerler içerisinde ise bu değeri "up_level" değişkeni ile eşitlemesini istedik. Bu fonksiyon tanımlama işleminden sonra daha sonra belirlemiş olduğumuz aykırı değerlerini bir "for" döngüsü ile çalıştırıp aykırı değerlerinden kurtulmuş olduk.

3.3.6 Modelleme

a) Modellemeye Hazırlık

Projemizin ön hazırlık aşamasını bitirdikten sonra makine öğrenmesi modellerimizi oluşturacağımız modelleme aşamasına geldik. Burada makine öğrenmesi algoritmaları oluşturup bu modellerin sonuçlarını gözlemleyeceğiz.

```
x = data.drop(["NewVolSales"],axis = 1)
y = data.NewVolSales
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.20, random_state = 0)
```

```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

```
x_train = pd.DataFrame(x_train)
x_test = pd.DataFrame(x_test)
x_train.columns = x.columns
x_test.columns = x.columns
```

Kod 13 – Modelleme Hazırlığı

Kod 13'de görülmekte olan kod satırlarında ilk önce hedef değişkenimiz veri kümemizden ayrılıp "y" değişkenine atandı ve geriye kalan veri kümemize ise "x" değeri verildi.

Veri kümemizi makine öğrenmesi algoritmasına sokabilmemiz için verimizi test ve eğitim kümeleri olarak ikiye ayırmamız gerekmektedir. Bu işlemi "train_test_split" komutuyla yapıyoruz. Bu komutun içinde bulunan "test_size" değişkenine "0.20" değerini atadık. Böylece verimiz eğitim için veri kümemizin %80'ini, test için ise %20'sini kullanacaktır.

Veri kümemizin standartlaştırma işleminden geçmesi gerektiğini daha önce bahsetmiştik. Bu işlemi "StandardScaler" metoduyla gerçekleştiriyoruz.

Veri kümemiz standartlaştıktan sonra "DataFrame" olma özelliğini kaybedip "Numpy" dizisine dönüşmektedir. Bu dönüşme işlemi ilerleyen işlemlerde bizim için sorunlar oluşturacağı için standardize edilmiş verimizi tekrar "DataFrame" haline getirip kolon isimlerini de tekrar atıyoruz.

b) Kullanılacak Algoritmalar ve Modelleme

1. Destek Vektör Regresyonu (Support Vector Regression) : İki sürekli değişken arasında doğrusal ilişkiyi inceleyen istatistiksel bir algoritmadır. Genellikle veriler arasında ilişkiyi açıklayabilecek bir doğru oluşturmaya çalışır [16].
2. AdaBoost Regresyonu (AdaBoost Regression): AdaBoost Regresyonu, orjinal veri kümesine bir bağımsız değişken yerleştirerek başlamaktadır. Bu yerleştirme işleminin ardından aynı veri kümesine bağımsız değişkenin ek kopyalarını yerleştirir ve değişkenlerin ağırlıklarını bu yerleştirmeler sonucu oluşan hatalara göre belirleyen bir algoritmadır [17].
3. Aşırı Gradyan Artırma Regresyonu (Extereme Gradient Boosting Regression, XGBRegression): Son derece verimli ve esnek olacak şekilde tasarlanmış optimize edilmiş bir gradyan eğilimli algoritmadır. Gradyan artırma çerçevesi altında makine öğrenmesi algoritmalarını uygulamaktadır. [18].
4. Rassal Orman Regresyonu (Random Forest Regression) : Veri kümesinin çeşitli alt örneklerine bir dizi sınıflandırma karar ağacına uygun hale getirmektedir. Tahmin doğruluğu ve aşırı öğrenmeyi kontrol etmek için ortalama değeri kullanan bir regresyon algoritmasıdır [19].
5. Karar Ağacı Regresyonu (Decision Tree Regression) : Karar ağaçları veriyi bir sinüs eğrisine uygun hale getirmek için ek olarak gürültülü gözlemler kullanmaktadır. Buna sonuç olarak sinüs eğrisine yaklaşan yerel doğrusal regresyonları öğrenmektedir [20].
6. Gradyan Eğilimli Ağaç Regresyon (Gradient Boosting Tree Regression): Gradyan Eğilimli Regresyon ileriye dönük aşamalı bir tarzda katkı modeli oluşturmaktadır. İsteğe bağlı türevlenebilir yapısı kayıp fonksiyonların optimizasyonuna izin verir. Her aşamada verilen kayıp fonksiyonunun negatif gradyanına bir regresyon ağacı yerleştirilmektedir [21].
7. K En Yakın Komşu Regresyonu (K – Nearest Neighbors Regression) : KNN regresyonu, sezgisel bir şekilde, aynı mahalledeki gözlemlerin ortalamasını alarak bağımsız değişkenler ile sürekli sonuç arasındaki ilişkiye yaklaşan parametrik olmayan bir yöntemdir [22].
8. Çok Katmanlı Algılayıcı Regresyonu (Multi Layer Perceptron Regression): Bir Çok Katmanlı Algılayıcı Regresyonu, en az üç düğüm katmanından oluşur: bir giriş katmanı, bir gizli katman ve bir çıktı katmanı. Giriş düğümleri dışında, her düğüm doğrusal olmayan bir aktivasyon işlevi kullanan bir nörondur. Çok Katmanlı Algılayıcı Regresyonu, eğitim için geri yayılım adı verilen denetimli bir öğrenme tekniğini kullanır. Çoklu katmanları ve doğrusal olmayan aktivasyonu, Çok Katmanlı Algılayıcı Regresyonu'nu doğrusal bir algılayıcıdan ayırır. Doğrusal olarak ayrılamayan verileri ayırt edebilir [23].
9. Polinominal Regresyon (Polynimal Regression): Belirtilen dereceye eşit veya daha düşük dereceye sahip özelliklerin tüm polinom kombinasyonlarıyla yeni bir özellik matrisi oluşturan regresyon algoritmasıdır [24].

Verilerimizi makine öğrenmesi modellerine hazır hale getirdik ve modelleme işlemine başlayabilmek için önümüzde bir engel kalmadığına göre modelleme işlemine başlayabiliriz.

```
svr = SVR(kernel = "rbf")
ada = AdaBoostRegressor()
xgb = XGBRegressor()
rfr = RandomForestRegressor()
dtr = DecisionTreeRegressor()
gbr = GradientBoostingRegressor()
knn = neighbors.KNeighborsRegressor()
mlp = MLPRegressor()
names = ["SVR","AdaBoostRegressor","XGBRegressor","RandomForestRegressor","DecisionTreeRegressor",
        "GradientBoostingRegressor","KNeighborsRegressor","MLPRegressor"]
models = [svr, ada, xgb, rfr, dtr, gbr, knn, mlp]
```

```
for name, clf in zip(names, models):
        clf.fit(x_train, y_train)
        y_pred = clf.predict(x_test)
        mse = mean_squared_error(y_test, y_pred)
        print("{}: mse score: {}".format(name, mse))
        r2 = r2_score(y_test, y_pred)
        print("{}: r2 score: {}".format(name, r2))
        mae = mean_absolute_error(y_test, y_pred)
        print("{}: mea score: {}".format(name, mae))
        mape = np.mean(abs((y_pred-y_test)/y_test))*100
        print("{}: mape :{}".format(name,mape))
        print("\n")
```

Kod 14 – Modelleme

Kod 14'de de görüldüğü üzere daha önceden "import" edilen makine öğrenmesi modellerinin her birini bir değere atadık. Bu değerlerden oluşan bir liste oluşturduk. Aynı zamanda bu değerlerin isimlerinden oluşan başka bir liste daha hazırladık.

Oluşturduğumuz bu değerleri bir "for" döngüsü içinde birer birer döndürerek elde ettiğimiz sonuçları yazdırmayı amaçladık.

c) Skorlama

Verimizi modelleme işlemlerine soktuktan sonra bu makine öğrenmesi işlemlerinden hangisini kullanmamız gerektiğine bu sonuçlara bakarak karar vereceğiz.

```
SVR: mse score: 4514270.289287768
SVR: r2 score: -0.2136858147814782
SVR: mea score: 1618.370475607868
SVR: mape :7.424368875771746

DecisionTreeRegressor: mse score: 1171345.0952380951
DecisionTreeRegressor: r2 score: 0.6850775794976527
DecisionTreeRegressor: mea score: 890.5238095238095
DecisionTreeRegressor: mape :4.275474079473435

AdaBoostRegressor: mse score: 937931.2715261905
AdaBoostRegressor: r2 score: 0.7478321397386034
AdaBoostRegressor: mea score: 672.0493617772523
AdaBoostRegressor: mape :3.177294634956641

GradientBoostingRegressor: mse score: 705574.320951617
GradientBoostingRegressor: r2 score: 0.8103025539597986
GradientBoostingRegressor: mea score: 677.9464266460242
GradientBoostingRegressor: mape :3.2453737191562624

[18:28:32] WARNING: src/objective/regression_obj.cu:152: reg:
XGBRegressor: mse score: 651887.171786717
XGBRegressor: r2 score: 0.8247366323826429
XGBRegressor: mea score: 655.1266741071429
XGBRegressor: mape :3.158407084573736

KNeighborsRegressor: mse score: 1161569.4571428578
KNeighborsRegressor: r2 score: 0.6877058123074562
KNeighborsRegressor: mea score: 856.3904761904764
KNeighborsRegressor: mape :4.004162246092586

RandomForestRegressor: mse score: 703098.226066666
RandomForestRegressor: r2 score: 0.8109682653694696
RandomForestRegressor: mea score: 642.9161904761901
RandomForestRegressor: mape :3.067240574026665

MLPRegressor: mse score: 434540878.8334973
MLPRegressor: r2 score: -115.82864932442973
MLPRegressor: mea score: 20756.23618338948
MLPRegressor: mape :99.91374171001569
```

Kod 15 – Skorlama

Oluşturduğumuz "for" döngüsü sayesinde bütün makine öğrenmesi algoritmalarının sonuçlarını bir arada görebiliyoruz. Bu algoritmalardan elde ettiğimiz metriklerden "R2 score'unu" hedef metriğimiz olarak ele alacağız.

Bu sonuçlar doğrultusunda en iyi algoritmalarımız "XGBRegressor" ,"RandomForestRegressor" ve "GradientBoostRegressor" algortimalarının en iyi değerlere sahip olarak görülmektedir.

```
poly_reg = PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(x_train)
```

```
lin_reg = lin_reg.fit(X_poly, y_train)
```

```
y_pred = lin_reg.predict(poly_reg.transform(x_test))
```

```
mse = mean_squared_error(y_test, y_pred)
print("MSE score",mse)
r2 = r2_score(y_test, y_pred)
print("r2 score",r2)
mea = mean_absolute_error(y_test, y_pred)
print("MAE score",mea)
```

```
MSE score 1231044.1768676755
r2 score 0.6690271607397775
MAE score 870.1663344047814
```

Kod 16 – Polinominal Regresyon

Polinominal Regresyon algoritmasının çalışma prensibi diğer algoritmalara göre değişik olduğunu için o algoritmayı "for" döngüsü içinde çalıştıramadık. Ama Polinominal Regresyon algoritmasından çıkacak sonucu öğrenmek için yine de bu algoritmayı oluşturup sonucuna baktık.

Polinominal Regresyon algoritmasını oluşturmak için eğitim kümemizi polinominal değerlere çevirmemiz gerekmektedir. Bu çevirme işlemini gerçekleştirdikten sonra Doğrusal Regresyon algoritması yardımıyla tahmin etme işlemini yapmaktayız. Elde ettiğimiz sonuç daha önceki sonuçlardan düşük olduğu için daha önceki algoritmalarla projemize devam edeceğiz.

3.3.7 Model Geliştirme İşlemleri

Modelleme işlemlerinden sonra seçmiş olduğumuz birbirine yakın sonuçlar veren algoritmaların parametrelerini optimize ederek maksimum başarı oranına ulaşmaya çalışacağız.

a) Rassal Orman Regresyonunun Parametre Optimizasyonu

```
parametresGrid_rfr = {"max_depth":[1,3,5,8,10],
                      "max_features":[2,5,10],
                      "min_samples_split":[5,10,20],
                      "n_estimators":[50,100,150,200]}
```

```
rfrr = GridSearchCV(rfr, parametresGrid_rfr, cv = 10, scoring="neg_mean_squared_error", refit = True,
                    n_jobs = -1, verbose = 2)
```

```
rfrr.fit(x_train, y_train)
```

```
Fitting 10 folds for each of 180 candidates, totalling 1800 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  58 tasks      | elapsed:    5.0s
[Parallel(n_jobs=-1)]: Done 194 tasks      | elapsed:   18.5s
[Parallel(n_jobs=-1)]: Done 397 tasks      | elapsed:   39.2s
[Parallel(n_jobs=-1)]: Done 680 tasks      | elapsed:  1.1min
[Parallel(n_jobs=-1)]: Done 1045 tasks      | elapsed:  1.8min
[Parallel(n_jobs=-1)]: Done 1490 tasks      | elapsed:  2.6min
[Parallel(n_jobs=-1)]: Done 1800 out of 1800 | elapsed:  3.1min finished
GridSearchCV(cv=10, estimator=RandomForestRegressor(), n_jobs=-1,
             param_grid={'max_depth': [1, 3, 5, 8, 10],
                         'max_features': [2, 5, 10],
                         'min_samples_split': [5, 10, 20],
                         'n_estimators': [50, 100, 150, 200]},
             scoring='neg_mean_squared_error', verbose=2)
```

```
rfrr.best_estimator_
```

```
RandomForestRegressor(max_depth=8, max_features=5, min_samples_split=5,
                      n_estimators=150)
```

Kod 17 – Parametre Optimizasyonu 1

Kod 17'de de görüldüğü üzere ilk olarak Rassal Orman Regresyonu algoritmasının parametrelerine belirli değerler atanmıştır. Bu değerler teker teker denenip elde edilen sonuçlardan başarı oranları yüksek olan parametre değerlerini kullanacağız.

Çapraz Doğrulama Algoritması

Çapraz Doğrulama Algoritması, genel anlamda aşırı-parametre uzayının tanımlanmış alt kümelerine dayanan kapsamlı bir aramadır. Aşırı parametreler, minimum değer, maksimum değer ve adım sayıları girilerek belirlenmektedir. Belirtme işlemi sonrasında algoritma bu değerler aralığında her bir kombinasyonu hesaplamaktadır. her kombinasyonun performansı belirli metriklerce değerlendirilip optimal sonuca ulaşmayı amaçlamaktadır [25].

Çapraz Doğrulama algoritmasının "cv" değerini 10 olarak belirledik. Bunun anlamı bu algoritma verimizi 10 parçaya bölüp geriye kalan 9 parçayla öğrenme işlemini gerçekleştirecektir. Bu öğrenme işlemi bütün parçalar, hem eğitim hem de test verisi olarak kullanıldıktan sonra bitecektir. Bu veri kümesinin 10 defa çalışması demektir.

İşlem bittikten sonra "best_estimator_" komutuyla algoritma da en yüksek başarıya ulaşan parametreleri görmüş olduk.

```
y_pred = rfrr.predict(x_test)
mse = mean_squared_error(y_test, y_pred)
print("Random Forest Mse score : ", mse)
r2 = r2_score(y_test, y_pred)
print("Random Forest R2 score",r2)
mape = np.mean(abs((y_pred-y_test)/y_test))*100
print("Random Forest mape score",mape)
```

```
Random Forest Mse score :  781819.2283909662
Random Forest R2 score 0.7898037010603491
Random Forest mape score 3.2996308991486347
```

Kod 18 – Skor 1

Parametre optimizasyonu sonrasında Rassal Orman Regresyonu algoritmasının başarı oranı 0.810'dan 0.781'e düşmüş olduğunu gözlemlenmiştir.

b) Gradyan Eğilimli Ağaç Regresyonu Parametre Optimizasyonu

```
parametresGrid_gbr = {"max_depth":[3,5,8,10],
                      "max_features":[2,5,10],
                      "min_samples_split":[10,50,100],
                      "n_estimators":[100,200,500,1000]}
```

```
gbrr = GridSearchCV(gbr, parametresGrid_gbr, cv = 10, scoring="neg_mean_squared_error", refit = True,
                    n_jobs = -1, verbose = 2)
```

```
gbrr.fit(x_train, y_train)
```

```
Fitting 10 folds for each of 144 candidates, totalling 1440 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  58 tasks      | elapsed:    2.7s
[Parallel(n_jobs=-1)]: Done 300 tasks      | elapsed:   18.1s
[Parallel(n_jobs=-1)]: Done 706 tasks      | elapsed:   45.0s
[Parallel(n_jobs=-1)]: Done 1140 tasks      | elapsed:  1.4min
[Parallel(n_jobs=-1)]: Done 1440 out of 1440 | elapsed:  1.7min finished
GridSearchCV(cv=10, estimator=GradientBoostingRegressor(), n_jobs=-1,
             param_grid={'max_depth': [3, 5, 8, 10], 'max_features': [2, 5, 10],
                         'min_samples_split': [10, 50, 100],
                         'n_estimators': [100, 200, 500, 1000]},
             scoring='neg_mean_squared_error', verbose=2)
```

```
gbrr.best_estimator_
```

```
GradientBoostingRegressor(max_features=2, min_samples_split=50,
                          n_estimators=200)
```

Kod 19 – Parametre Optimzasyonu 2

Kod 19'de de görüldüğü üzere ilk olarak Gradyan Eğilimli Ağaç Regresyonu algoritmasının parametrelerine belirli değerler atanmıştır. Bu değerler teker teker denenip elde edilen iyi sonucu verenler parametreleri kullanacağız.

Çapraz Doğrulama algoritmasının "cv" değerini 10 olarak belirledik. Bunun anlamı bu algoritma verimizi 10 parçaya bölüp geriye kalan 9 parçayla öğrenme işlemini gerçekleştirecektir ve bu öğrenme işlemi bütün parçalar hem eğitim hem de test verisi olarak kullanıldıktan sonra bitecektir. Bu veri kümesinin 10 defa çalışması demektir.

İşlem bittikten sonra "best_estimator_" komutuyla algoritma da en yüksek başarıya ulaşan parametreleri görmüş olduk.

```
y_pred = gbrr.predict(x_test)
mse = mean_squared_error(y_test, y_pred)
print("Gradient Boost Mse score : ", mse)
r2 = r2_score(y_test, y_pred)
print("Gradient Boost R2 score",r2)
mape = np.mean(abs((y_pred-y_test)/y_test))*100
print("Gradient Boost mape score",mape)
```

```
Gradient Boost Mse score :  616546.0149525615
Gradient Boost R2 score 0.8342382922255119
Gradient Boost mape score 3.221548868160009
```

Kod 20 – Skor 2

Parametre optimizasyonu sonrasında Gradyan Eğilimli Ağaç Regresyonu algoritmasının başarısı 0.810'dan 0.834'e arttığı gözlemlenmiştir.

c) Aşırı Gradyan Artırma Regresyonu Parametre Optimizasyonu

```
parametresGrid_xgb = {"nthread":[4],"objective":["reg:linear"],
                      "learnin_rate":[0.03,0.05,0.07], "max_depth" : [5,6,7],
                      "min_child_weight":[4,5,6], "silent":[1],
                      "subsample":[0.7], "colsample_bytree":[0.7],
                      "n_estimators":[100,500,1000]}
```

```
xgbr = GridSearchCV(xgb, parametresGrid_xgb, cv = 10, scoring="neg_mean_squared_error", refit = True,
                    n_jobs = -1, verbose = 2)
```

```
xgbr.fit(x_train, y_train)
```

```
Fitting 10 folds for each of 81 candidates, totalling 810 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  33 tasks      | elapsed:   11.6s
[Parallel(n_jobs=-1)]: Done 154 tasks      | elapsed:  1.0min
[Parallel(n_jobs=-1)]: Done 357 tasks      | elapsed:  2.6min
[Parallel(n_jobs=-1)]: Done 640 tasks      | elapsed:  4.9min
[Parallel(n_jobs=-1)]: Done 810 out of 810 | elapsed:  6.5min finished
GridSearchCV(cv=10, estimator=XGBRegressor(), n_jobs=-1,
             param_grid={'colsample_bytree': [0.7],
                         'learnin_rate': [0.03, 0.05, 0.07],
                         'max_depth': [5, 6, 7], 'min_child_weight': [4, 5, 6],
                         'n_estimators': [100, 500, 1000], 'nthread': [4],
                         'objective': ['reg:linear'], 'silent': [1],
                         'subsample': [0.7]},
             scoring='neg_mean_squared_error', verbose=2)
```

```
xgbr.best_estimator_
```

```
XGBRegressor(colsample_bytree=0.7, learnin_rate=0.03, max_depth=7,
             min_child_weight=4, nthread=4, silent=1, subsample=0.7)
```

Kod 21 – Parametre Optimizasyonu 3

Kod 21'de de görüldüğü üzere ilk olarak Aşırı Gradyan Artırma Regresyonu algoritmasının parametrelerine belirli değerler atanmıştır. Bu değerler teker teker denenip elde edilen iyi sonucu verenler parametreleri kullanacağız.

Çapraz Doğrulama algoritmasının "cv" değerini 10 olarak belirledik. Bunun anlamı bu algoritma verimizi 10 parçaya bölüp geriye kalan 9 parçayla öğrenme işlemini gerçekleştirecektir ve bu öğrenme işlemi bütün parçalar hem eğitim hem de test verisi olarak kullanıldıktan sonra bitecektir. Bu veri kümesinin 10 defa çalışması demektir.

İşlem bittikten sonra "best_estimator_" komutuyla algoritma da en yüksek başarıya ulaşan parametreleri görmüş olduk.

```
y_pred = xgbr.predict(x_test)
mse = mean_squared_error(y_test, y_pred)
print("XGBoost Mse score : ", mse)
r2 = r2_score(y_test, y_pred)
print("XGBoost R2 score",r2)
mape = np.mean(abs((y_pred-y_test)/y_test))*100
print("XGBoost mape score",mape)
```

```
XGBoost Mse score :  544149.6867557707
XGBoost R2 score 0.8537024339237202
XGBoost mape score 2.8919764391273075
```

Kod 22 – Skor 3

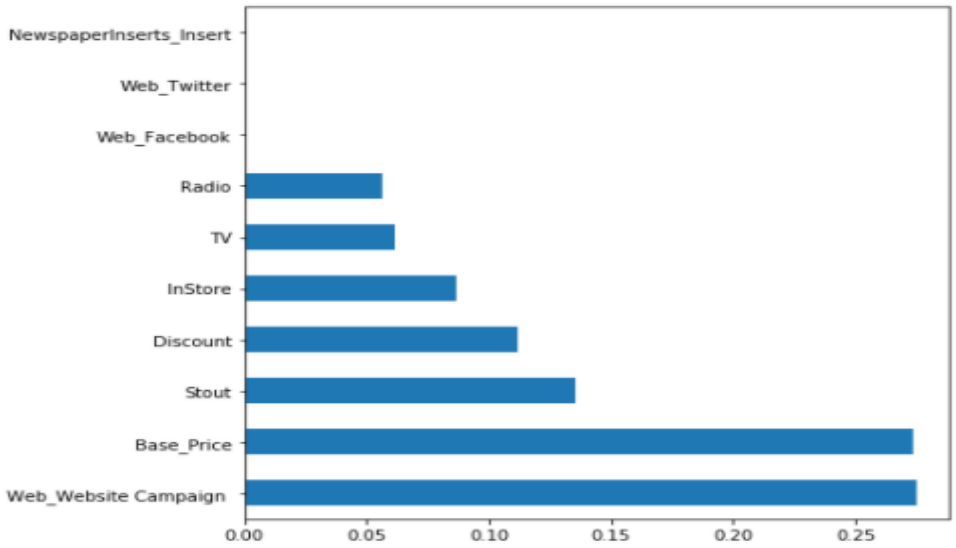
Parametre optimizasyonu sonrasında Aşırı Gradyan Artırma Regresyonu algoritmasının başarısı 0.824'den 0.853'3 arttığını gözlemlenmiştir.

Yapılan parametre optimizasyonlarından sonra elde edilen R^2 sonuçları;

Aşırı Gradyan Artırma Regresyonu Algoritması : 0.853

Gradyan Eğilimli Ağaç Regresyonu Algoritması : 0.834

Rassal Orman Regresyonu : 0.789 sonuçlarını vermişlerdir. En iyi çalışan algoritmamız Aşırı Gradyan Artırma Regresyonu algoritması oluştur. Projemize Aşırı Gradyan Artırma Regresyonu algoritması ile devam edilecektir.




```
xgbr.feature_importances_
```

```
array([0.27359176, 0.05622134, 0.08695688, 0.11189445, 0.06120925,
       0.13536787, 0.        , 0.        , 0.27475846, 0.        ],
      dtype=float32)
```

Kod 23 – Öznitelik Önem Katsayıları

Aşırı Gradyan Artırma Regresyonu Algoritmasının öznitelik önem değerleri Kod 23'teki gibidir.

3.3.9 Makine Öğrenmesi Algoritmalarının Performanslarının Karşılaştırılması

Araştırma boyunca elde ettiğimiz makine öğrenmesi performanslarının karşılaştırmasını yapılacaktır. Bu sayede veri kümemizin hangi algoritmalarda daha iyi çalıştığını gözlemleme şansımız olacaktır.

Makine Öğrenmesi Algoritmaları	**R Kare skorları**
Polinominal Regresyon (Knime)	0.819
Gradyan Eğilimli Ağaç Regresyonu (Knime)	0.939
Rassal Orman Regresyonu (Knime)	0.702
Destek Vektör Regresyonu	-0.213
AdaBoost Regresyonu	0.747
Aşırı Gradyan Artırma Regresyonu	0.853
Rassal Orman Regresyonu	0.818
Karar Ağacı Regresyonu	0.722
Gradyan Eğilimli Ağaç Regresyon	0.839
K En Yakın Komşu Regresyonu	0.687
Çok Katmanlı Algılayıcı Regresyonu	-115.870
Polinominal Regresyon	0.669

Tablo 1 – Performans Karşılatırması

Bütün makine öğrenmesi algoritmalarının performansların en iyi olduğu değerleri alarak bu tablo hazırlanmıştır. Tablo 1'de de görüldüğü üzere algoritmalar arasında en iyi performans gösteren algoritma Knime Programında oluşturmuş olduğumuz Gradyan Eğilimli Ağaç Regresyonu'dur fakat bu sonuca ulaşmak için yaptığımız öznitelik seçimi işlemlerinde veri kümemizden çıkan değişkenlerimiz olduğu için bu performansı en başarılı performans olduğunu söyleyemeyiz. Bir sonraki

en iyi performans ve Gradyan Eğilimli Ağaç Regresyonu algoritmasını geliştirilerek oluşturulmuş olan Aşırı Gradyan Artırma Regresyonu'nun veri kümemize en iyi uyan algoritma olduğu söylenebiliriz. Ayrıca R^2 skorları arasında eksi değerler bulunmaktadır. Bu eksi değerler genellikle o algoritmanın başarısının, bütün tahmin değerlerine ortalama değer verilerek oluşturulacak bir değerlendirmeden daha kötü olduğu şeklinde yorumlanmaktadır.

3.3.10 Fiyat Esnekliği (Price Elasticity)

Esneklik, pazarlama aktivitelerinin bizim satışımıza olan olumlu veya olumsuz olarak etkileri nelerdir sorusunu cevaplamaktadır. Bu işlem diğer pazarlama unsurları içinde geçerlidir. Doğrusal Regresyon modellerinde "beta*(x.mean/y.mean)" olarak hesaplanmaktadır[26].

Bu hesaplama işlemini yapamıyoruz çünkü Aşırı Gradyan Artırma Regresyonu algoritmasından beta katsayılarını alamıyoruz bunun için başka bir yöntemle beta katsayılarımızı oluşturacağız.

a) Sıradan En Küçük Kareler Regresyonu (Ordinary Least Squares Regression)

Sıradan en küçük kareler regresyonu, bir veya birden fazla bağımsız değişken ile bir bağımlı değişken arasındaki ilişkiyi tahmin eden istatistiksel bir analiz yöntemidir. Yöntem; düz bir çizgi olarak yapılandırılmış bağımlı değişkenin gözlemlenen ve tahmin edilen değerleri arasındaki farkların kareleri toplamını en az indirerek ilişkiyi tahmin etmektedir [27].

Modelimizin beta katsayılarına erişmek için Aşırı Gradyan Artırma Regresyonu modeliyle tahmin ettiğimiz değişkenleri Sıradan En Küçük Kareler Regresyonu modeline sokarak elde edeceğiz.

Sıradan En Küçük Kareler Regresyonu modelini çalıştırmak için "stastmodel" kütüphanesini kullanacağız.

```
y_pred = xgbr.predict(x_train)
```

```
model_xgb = sm.ols(formula="y_pred~x_train", data=x_train).fit()
print(model_xgb.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 y_pred   R-squared:                       0.838
Model:                            OLS   Adj. R-squared:                  0.815
Method:                 Least Squares   F-statistic:                     37.18
Date:                Sun, 07 Mar 2021   Prob (F-statistic):           1.68e-24
Time:                        19:02:46   Log-Likelihood:                -640.66
No. Observations:                  83   AIC:                             1303.
Df Residuals:                      72   BIC:                             1330.
Df Model:                          10
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   2.001e+04     64.169    311.844      0.000    1.99e+04    2.01e+04
x_train[0] -1028.6312     70.573    -14.575      0.000   -1169.316    -887.946
x_train[1]   -55.2848     68.275     -0.810      0.421    -191.388      80.819
x_train[2]   392.8915     92.936      4.228      0.000     207.627     578.156
x_train[3]   132.6426     93.626      1.417      0.161     -53.998     319.283
x_train[4]   133.2756     72.396      1.841      0.070     -11.043     277.594
x_train[5]  -523.8836     72.209     -7.255      0.000    -667.829    -379.938
x_train[6]    81.9078     66.814      1.226      0.224     -51.283     215.098
x_train[7]    34.9251     70.540      0.495      0.622    -105.693     175.543
x_train[8]  -309.2659     73.556     -4.205      0.000    -455.897    -162.635
x_train[9]    57.2309     65.685      0.871      0.386     -73.710     188.172
==============================================================================
Omnibus:                        4.000   Durbin-Watson:                   1.808
Prob(Omnibus):                  0.135   Jarque-Bera (JB):                3.343
Skew:                           0.352   Prob(JB):                        0.188
```

Kod 24 – Sıradan En Küçük Kareler Regresyonu

Sıradan En Küçük Kareler Regresyonu modelini oluşturmak için "x_train" değişkenimizi Aşırı Gradyan Artırma Regresyonu ile tahmin ettik ve model içinde "y_pred" değişkeni olarak bu değeri girdik. "x" değişkeni olarak "x_train" değerini girdik ve sonra olarak "data" değişkenine de "x_train" verisi atayıp modelimizi "fit" ettik. Artık beta katsayılarını elde ettiğimize göre esneklik hesaplamasını yapabiliriz.

b) Esneklik hesaplama

```
[114] def get_params(x,y):
          model = sm.ols(formula="y~x", data=x).fit()
          coef = model.params
          coef = coef[1:]
          return coef
```

```
def elasticty(x,y,get_params):
      x_mean = x.mean()
      elasticity = np.column_stack([coef, mean]).tolist()
      elasticity = pd.DataFrame(elasticity)
      elasticity.columns = (["Coeffiecents","Mean"])
      elasticity["elasticty_value"] = elasticity["Coeffiecents"]*(elasticity["Mean"]/y.mean())
      elasticity.head(20)
      return elasticity
```

+ Code + Text

Kod 25 – Esneklik Hesaplama

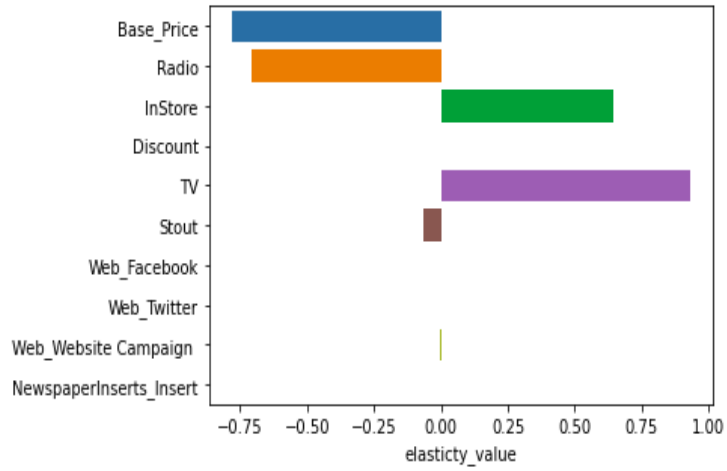
Kod 25'te görüldüğü üzere iki adet fonksiyon oluştarak önce veri kümemizde bulunan değişkenlerin beta katsıyalarını hesapladık. İkinci fonksiyonda ise bir önceki fonksiyonda hesaplamış olduğumuz beta katsayılarını kullanarak değişkenlerimizin esneklik değerlerinin hesaplanmasını sağladık.

c) Esneklik Değerleri ve Grafiği

```
elasticty_data = elasticty(x,y,coef)
elasticty_data
```

	Coeffiecents	Mean	elasticty_value
0	-1028.631219	15.323059	-0.781405
1	-55.284830	257.528846	-0.705835
2	392.891500	32.918567	0.641187
3	132.642642	0.022059	0.000145
4	133.275582	141.009774	0.931689
5	-523.883557	2.545966	-0.066124
6	81.907806	0.038462	0.000156
7	34.925065	0.038462	0.000067
8	-309.265917	0.057692	-0.000885
9	57.230914	0.057692	0.000164

```
sns.barplot(x=elasticty_data["elasticty_value"],
            y=x.columns, data=elasticty_data)
plt.show()
```



Kod 26 – Esneklik Sonuçları

4. Sonuçların Değerlendirilmesi

Kod 26'da da görüldüğü üzere değişkenlerimizin Esneklik değerleri gözükmektedir. Bu değerler bu değişkenlerde yapılacak %1'lik artışın fiyatlar üzerinde nasıl etkisi olacağını bize göstermektedir.

- Base_Price değişkeninin Esneklik değeri -0.781 olarak hesaplanmıştır. Bu değerin anlamı bu birim fiyatta yapılacak olan %1'lik artış bizim toplam satışımızı %0.78 oranında düşüreceği anlamına gelmektedir
- Radio değişkenin Esneklik değeri -0.705 olarak hesaplanmıştır. Bu değerin anlamı radyoya pazarlama kanalına yapılacak %1'lik artışın toplam satışımızı %0.70 oranında düşüreceği anlamına gelmektedir.
- Instore değişkenin Esneklik değeri 0.641 olarak hesaplanmıştır. Bu değerin anlamını mağaza içi pazarlama harcamalarında yapacağımız %1'lik artışın toplam satışımızı %0.64 oranında arttıracağı anlamına gelmektedir.
- TV değişkenin Esneklik değeri 0.931 olarak hesaplanmıştır. Bu değerin anlamı televizyon pazarlama kanalına yapılacak %1'lik artışın toplam satımızı %0.931 oranında arttıracağı anlamına gelmektedir.

Bu veriler ışığında bu satışları yapmış olan satıcıya ilerleyen dönemlerdeki pazarlama harcamalarında ürünün birim fiyatında bir miktar indirime gidilmesi, Radyo kanalına yaptığı pazarlama harcamalarını kısmasını tavsiye edebiliriz. Radyo kanalından kısılan pazarlama bütçesini televizyon ve mağaza içi promosyonlara yönelik kullanması da bu sonuçlardan elde edebileceğimiz çıkarımlardan biridir. Aynı zamanda diğer küçük etkileri olan verilere bakıldığında "Stout" değişkeninde eksi değer gözlenmektedir. Bu nedenle mağaza stok sayılarının artırılmaması toplam satışımızın artmasında etkili olabilir. Ve son olarak elde edilen web reklamlarının çıktılarına bakılarak web site üzerinden yapılan pazarlama harcamaları diğer web kanallarından daha olumlu sonuçlar getirdiği söylenebilir.

KAYNAKÇA

[1] Gian M. Fylgoni << The Journal of Advertising Research vol 58 issue 4 >> 1 Aralık 2018. [Çevrimiçi] – Ulaşılabilir: http://www.journalofadvertisingresearch.com/content/58. Sayı /4. Makale /sayfa 390

[2] Frances, P. H. (2005). <<The use of econometric models for policy simulation in marketing. Journal of Marketing Research,>> . [Çevrimiçi] – Ulaşılabilir; https://journals.sagepub.com/doi/abs/10.1509/jmkr.42.1.4.56891 42. Sayı, sayfa 4-14.

[3] Neff, J. (2007). << Representing PR in the Marketing Mix A Study on Public Relations Variables in Marketing Mix Modeling),>> [Çevrimiçi] – Ulaşılabilir; https://instituteforpr.org/wp-content/uploads/BG_SmithKetchum.pdf 78(10), sayfa 8

[4] Micheal J. Wolfe, John C. Crotts << Marketing Mix Modeling for the Tourism Industry'dir" >> 16 Nisan 2011. [Çevrimiçi] – Ulaşılabilir file; // https://www.tandfonline.com/doi/abs/10.1080/15980634.2011.11434633, sayfa 2

[5] Micheal J. Wolfe, John C. Crotts << Marketing Mix Modeling for the Tourism Industry'dir" >> 16 Nisan 2011. [Çevrimiçi] – Ulaşılabilir file; // https://www.tandfonline.com/doi/abs/10.1080/15980634.2011.11434633

[6] Bagher Asgarnezhad Nouri ve Milad Soltani << Evaluating the Effect of Tourism Marketing Mix on Buying Holiday Homes in Cyprus>> 7 Eylül 2015. [Çevrimiçi] – Ulaşılabilir; http://citeseerx.ist.psu.edu/viewdoc-/download?doi= 10.1.1.1025.1031&rep=rep1&type=pdf

[7] Nielsen Holdings << MARKETING MIX MODELING WHAT MARKETING PROFESSIONALS NEED TO KNOW >> Mart 2014. [Çevrimiçi] – Ulaşılabilir; https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/marketing-mix-modeling-what-marketers-need-to-know.pdf , sayfa 8-14

[8] Şadi Evren ŞEKER << CRISP-DM: Endüstriler Arası Standart İşleme – Veri Madenciliği için (Cross Industry Standard Processing – Data Mining) >> Temmuz 2018. [Çevrimiçi] – Ulaşılabilir; https://ybsansiklopedi.com/wp-content/uploads/2018/08/crispdm.pdf , sayfa 11-13

[9] https://www.kaggle.com/veer06b/marrket-mix-dataset

[10] İstatistik Merkezi << SPEARMAN KORELASYON ANALİZİ VE TEMEL PROBLEMLERİ >> [Çevrimiçi] – Ulaşılabilir; https://www.istmer.com/spearman-korelasyon-analizi-ve-temel-problemleri/

[11] M. Fevzi ESEN ve Mehpare Timor << ÇOK DEĞİŞKENLİ AYKIRI DEĞER TESPİTİ İÇİN KLASİK VE DAYANIKLI MAHALANOBİS UZAKLIK ÖLÇÜTLERİ: FİNANSAL VERİ İLE BİR UYGULAMA >> 21 Ağustos 2019 sayfa – 2 [Çevrimiçi] – Ulaşılabilir; :http://161.9.143.160:8080/bitstream/handle/20.500.12627/497/%C3%87OK%20DE%C4%9E%C4%B0%C5%9EKENL%C4%B0%20AYKIRI%20DE%C4%9EER%20TESP%C4%B0T%C4%B0%20%C4%B0%C3%87%C4%B0N%20KLAS%C4%B0K%20VE%20DAYANIKLI%20MAHALANOB%C4%B0S%20UZAKLIK%20%C3%96L%C3%87%C3%9CTLER%C4%B0%20F%C4%B0NANSAL%20VER%C4%B0%20%C4%B0LE%20B%C4%B0R%20UYGULAMA%20.pdf?sequence=1&isAllowed=y

[12] Emre Rençberoğlu << Fundamental Techniques of Feature Engineering for Machine Learning >> 1 Nisan 2019. [Çevrimiçi] – Ulaşılabilir; https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114

[13] Matthew Kramer << R2 STATISTICS FOR MIXED MODELS >> 2005 [Çevrimiçi] – Ulaşılabilir; https://pdfs.semanticscholar.org/1b57/e7fa9a8b8f6921cc26eb33b0b70afc0a9849.pdf, sayfa 3

[14] Deliktaş, B. Türker, H.T., Coşkun, H., Bikçe, M.,Özdemir, E. 2005. Genetik Algoritma Parametrelerinin Betonarme Kiriş Tasarımı Üzerine Etkisi. Bölgesel Jeoloji-Tektonik ve Sismotektonik Deprem Kaynak Mekanizmaları ve Dalga Yayınımı Sempozyumu, sayfa 23-25 Mart, Kocaeli, sayfa 46-54

[15] Yanıkoğlu, H., Özkara, E., Yüceer, M. 2010. Kinetik Model Parametrelerinin Belirlenmesinde Kullanılan Optimizasyon Tekniklerinin Kıyaslanması. 9. Ulusal Kimya Mühendisliği Kongresi(UKMK-9), 22-25 Haziran, Ankara.

[16] https://heartbeat.fritz.ai/support-vector-regression-in-python-using-scikit-learn-89cc18e933b7#:~:text=Support%20vector%20regression%20(SVR)%20is,relationship%20between%20two%20continuous%20variables.&text=In%20the%20realm%20of%20machine,other%20common%20and%20popular%20algorithms.

[17] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html

[18] https://xgboost.readthedocs.io/en/latest/index.html

[19] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[20] https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

[21] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

[22] https://bookdown.org/tpinto_home/Regression-and-Classification/k-nearest-neighbours-regression.html

[23] https://en.wikipedia.org/wiki/Multilayer_perceptron

[24] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html

[25] Iwan Syarif, Adam Prugel-Bennett, Gary Wills << SVM Parameter Optimization Using Grid Search and Genetic Algorithm to Improve Classification Performance >> Aralık 2016 [Çevrimiçi] – Ulaşılabilir; https://core.ac.uk/download/pdf/295538475.pdf, sayfa 2

[26] Deepanshu Bhalla << A COMPLETE GUIDE TO MARKETING MIX MODELING >> Aralık 2019. [Çevrimiçi] – Ulaşılabilir: https://www.listendata.com/2019/09/marketing-mix-modeling.html

[27] Dudley L. Poston Jr . << Ordinary Least Squares Regression >> [Çevrimiçi] – Ulaşılabilir: https://www.encyclopedia.com/social-sciences/applied-and-social-sciences-magazines/ordinary-least-squares regression#:~:text=Ordinary%20least%20squares%20-(OLS)%20regression,and%20predicted%20values%20of%20the

[28] Ş. E. ŞEKER, «Her Seviyeye Uygun Uçtan Uca Veri Bilimi, Knime ile,» UDEMY, [Çevrimiçi] – Ulaşılabilir:: https://www.udemy.com/veribilimi/learn/v4/content. [Erişildi: Şubat 2021].

[29] Ş. E. ŞEKER, «Python ile Makine Öğrenmesi,» Udemy, [Çevrimiçi] – Ulaşılabilir; https://www.udemy.com/makine-ogrenmesi/learn/v4/content. [Erişildi: Şubat 2021].