**Araba Tahmin Projesi Rapor Dökümanı**

İçindekiler

[Araba Fiyat Tahmini Projesi Tanımı 2](#_Toc469696682)

[1. Amaç 2](#_Toc469696683)

[2. Girdiler 2](#_Toc469696684)

[3. Mantığı 2](#_Toc469696685)

[4. Microsoft Azure Machine Learning Studio’da Model Oluşturma 2](#_Toc469696686)

[5. Oluşturulan Modeli Web Servisi Olarak Yayınlama 7](#_Toc469696687)

[6. Projenin Microsoft Azure Machine Learning Studio Ortamında Test Edilmesi 10](#_Toc469696688)

[7. Programın Kaynak Kodları 11](#_Toc469696689)

[8. Proje de kullanılan referanslar ve kütüphaneler 14](#_Toc469696690)

[9. Program Ekran Çıktıları 14](#_Toc469696691)

[9.1. Ekran Çıktısı 1 14](#_Toc469696692)

[9.2. Ekran Çıktısı 2 15](#_Toc469696693)

[9.3. Ekran Çıktısı 3 15](#_Toc469696694)

[9.4. Ekran Çıktısı 4 16](#_Toc469696695)

Araba Fiyat Tahmini Projesi Tanımı

Makine öğrenmesi Linear Regrasyon metodu ile model analiz edildi ve veri setinden yararlanılarak model eğitildi. Microsoft Azure Machine Learning Studio ile oluşturulan proje modeli C# .Net ortamında kullanıldı.Program içerisinde girilen özelliklere göre fiyat tahmin etme işlemi yapıldı. Bu “Araba Fiyat Tahmini Projesi Rapor Dökümanı” da bu modelin içeriği ve çalışması anlatıldı.

1. Amaç

Araba Fiyat Tahmini Projesi, arabaların fiyatlarını sahip oldukları özelliklere göre mantıklı bir şekilde tahmin etmeye yarar.

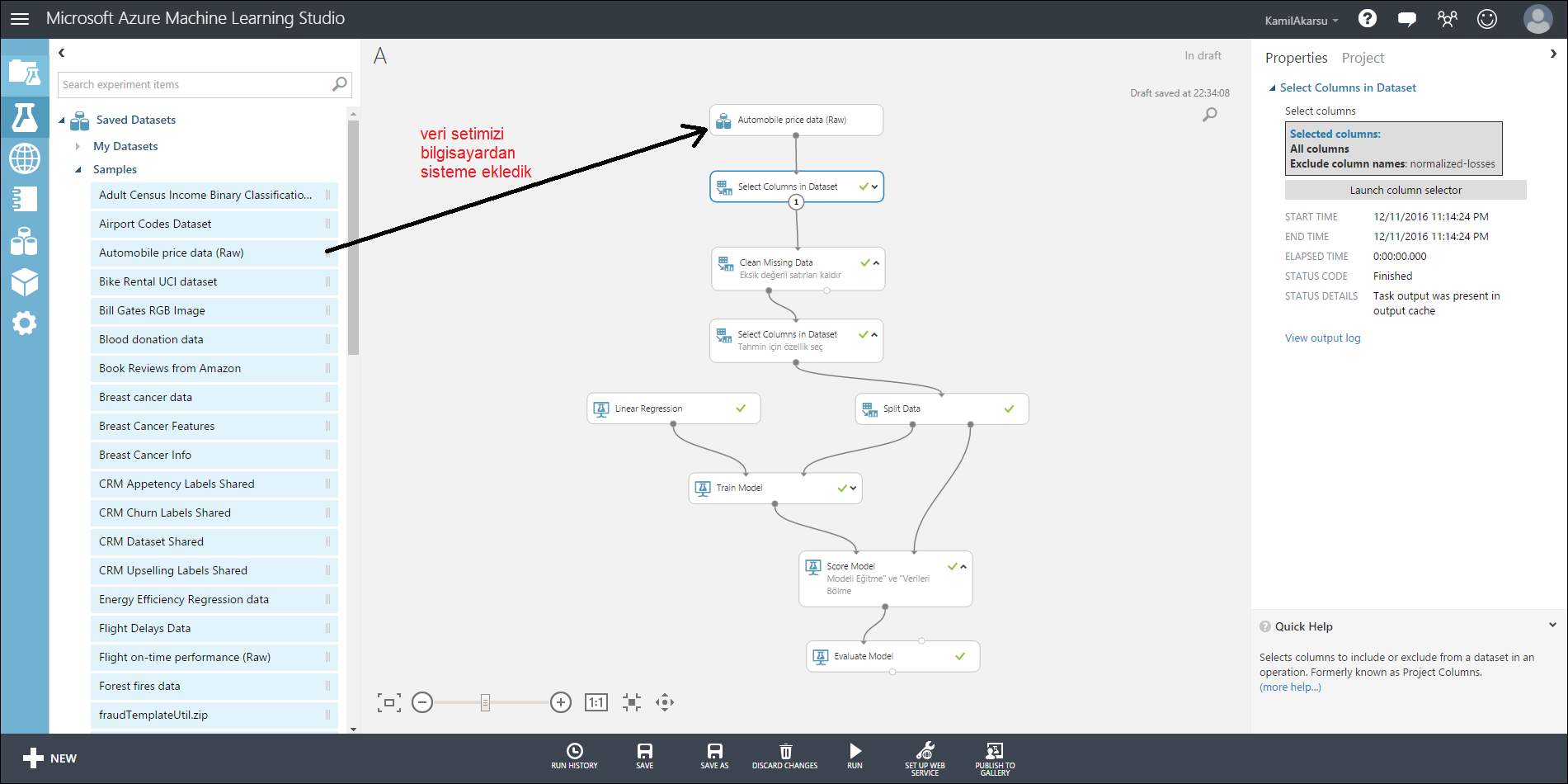
2. Girdiler

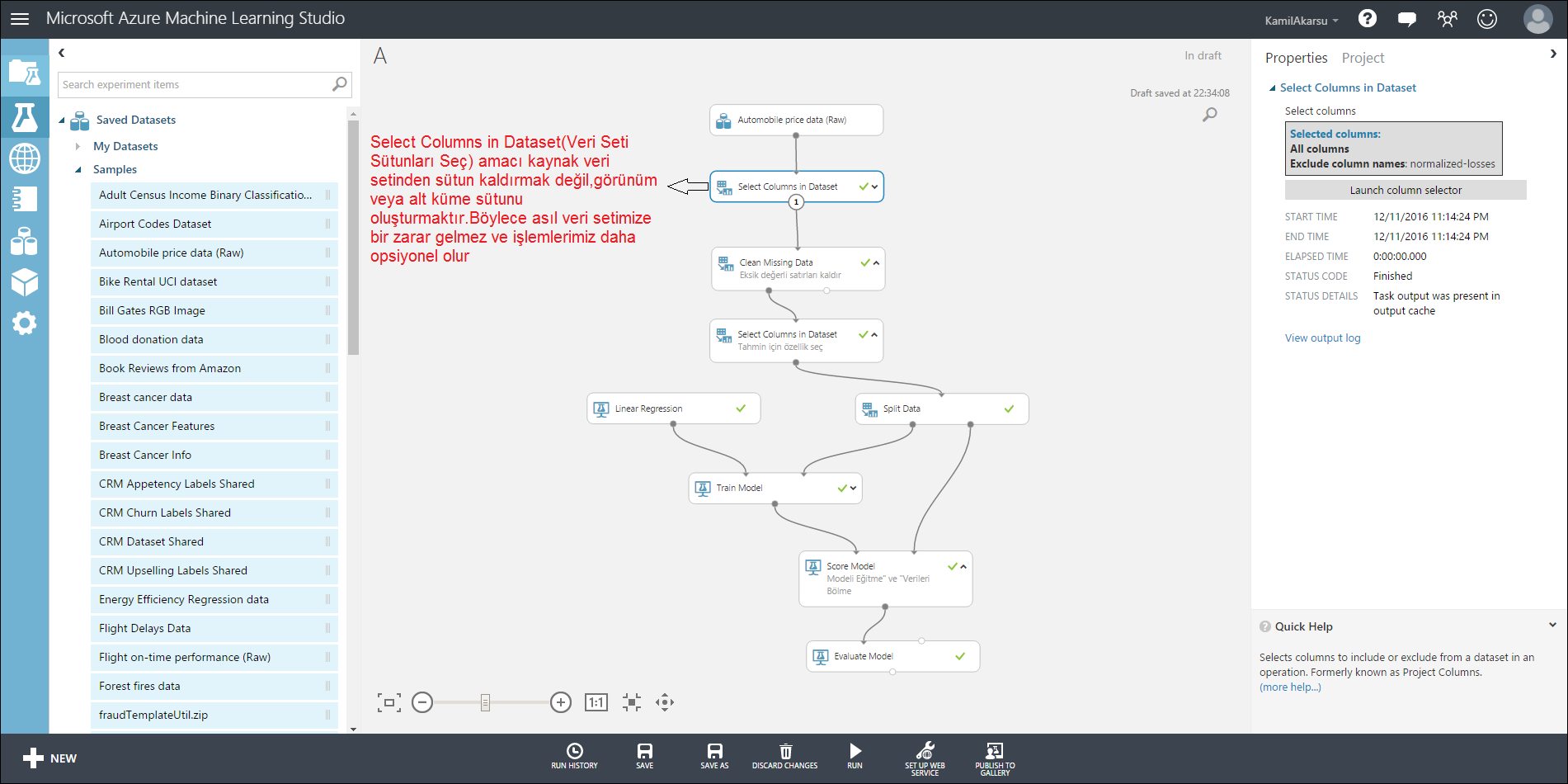
Arabaların teknik özellikleri

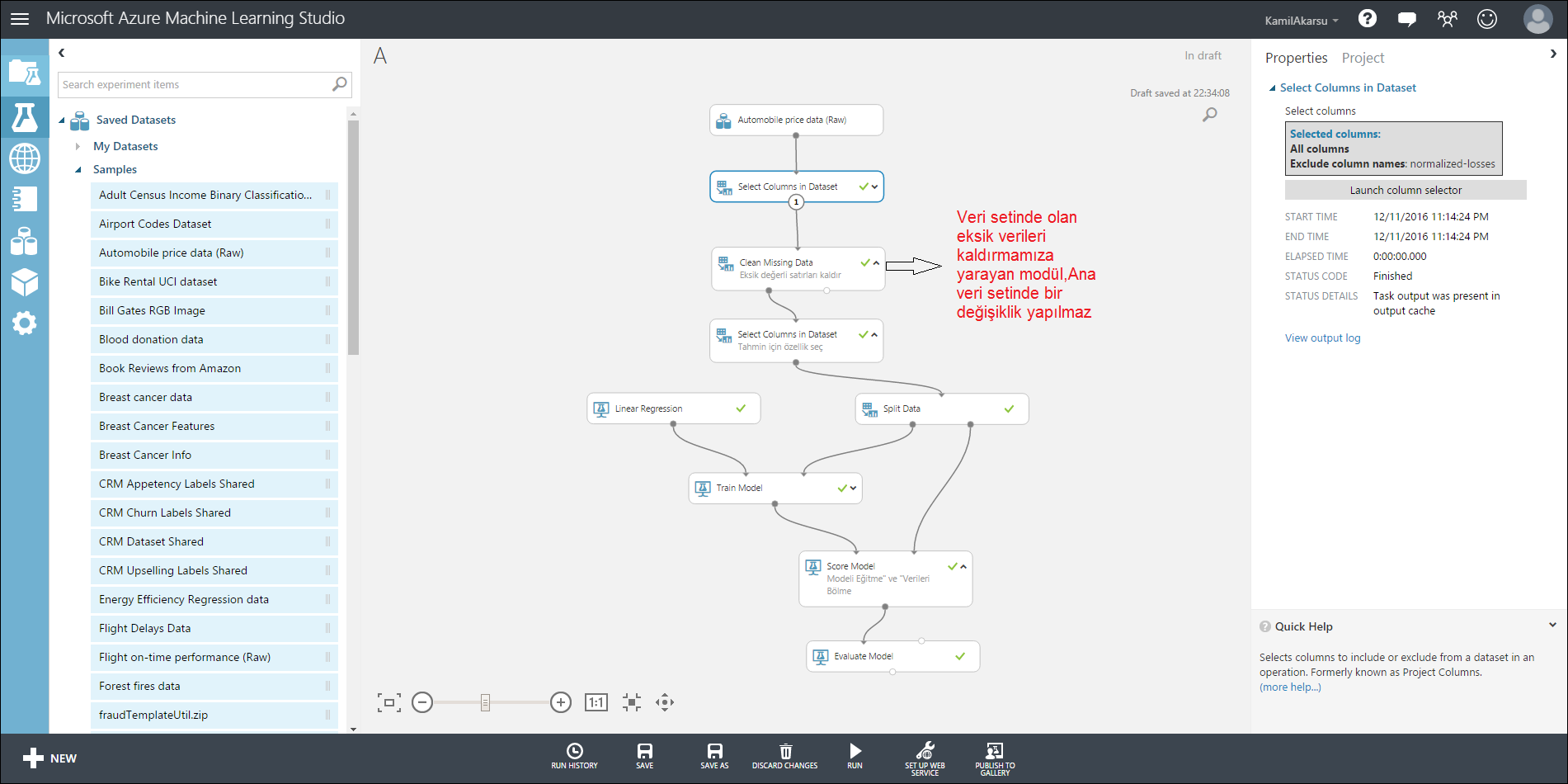
3. Mantığı

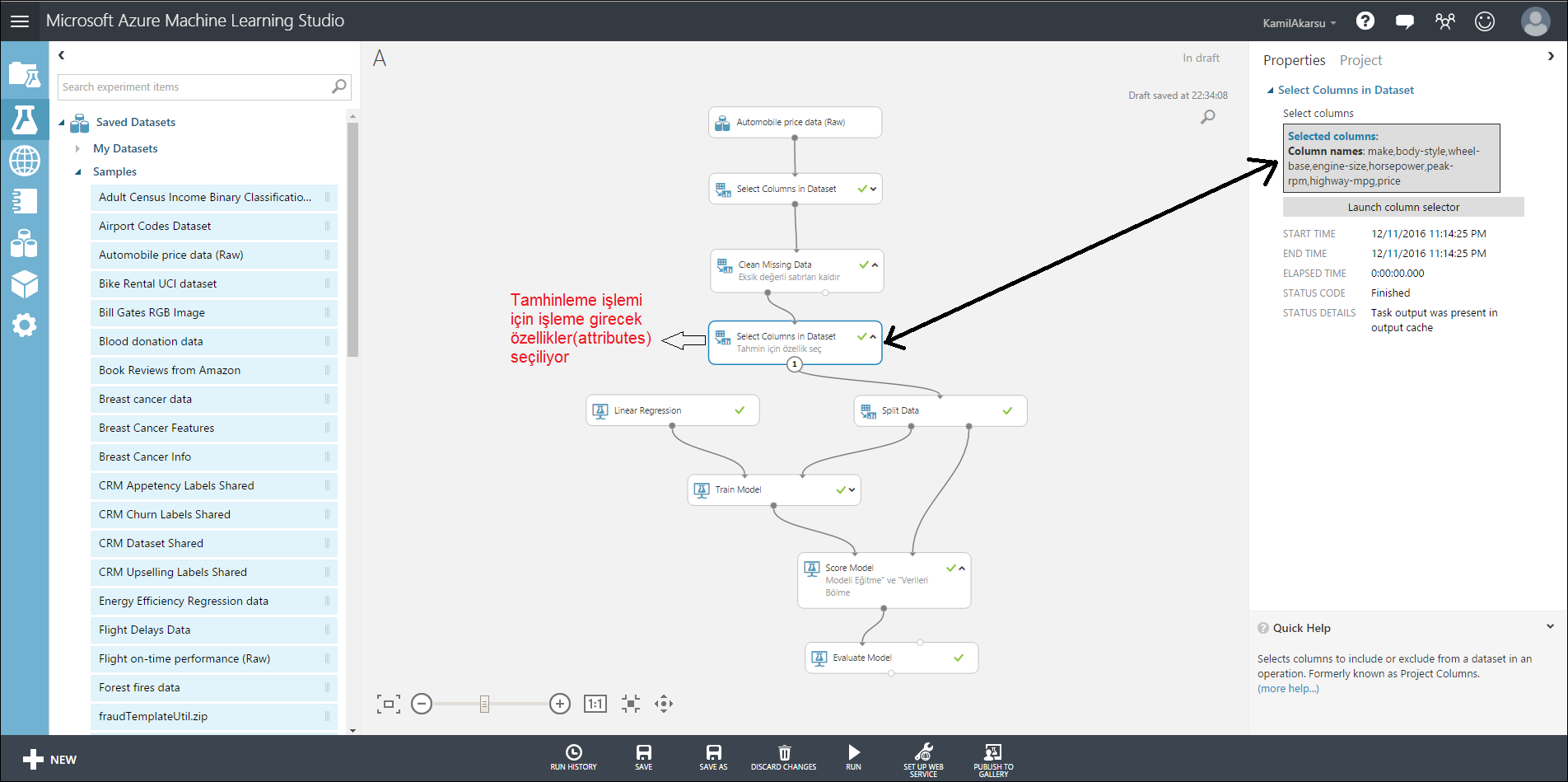
Microsoft Azure Machine Learning Studio’da oluşturulan proje modeli API Key ile programda kullanıldı. Böylece kullanıcının girmiş olduğu araba özellikleri Web Servisine gönderildi. Gönderilen özellikler doğrultusunda programa tahmini araba fiyatı döndürüldü.

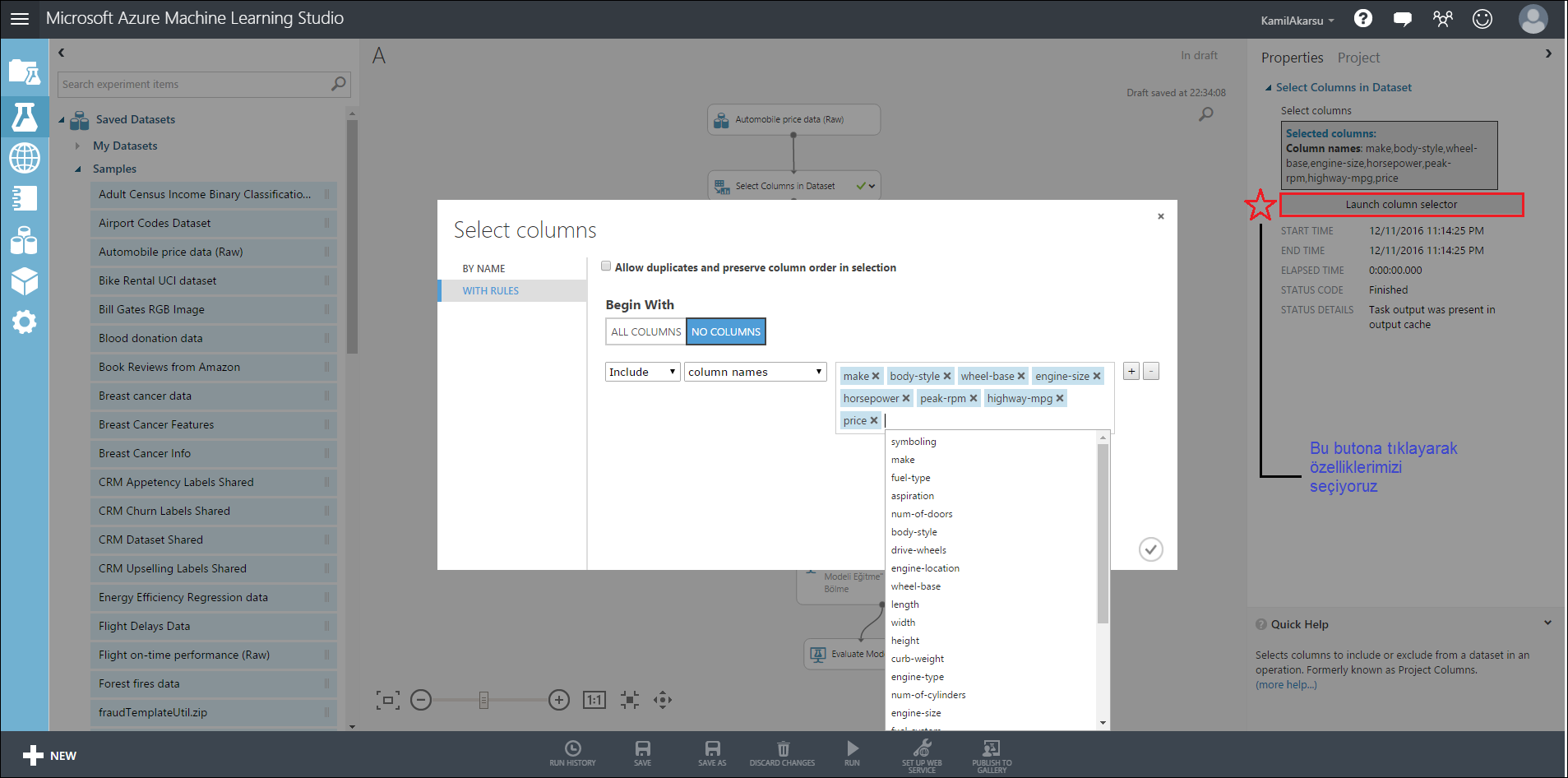
4. Microsoft Azure Machine Learning Studio’da Model Oluşturma

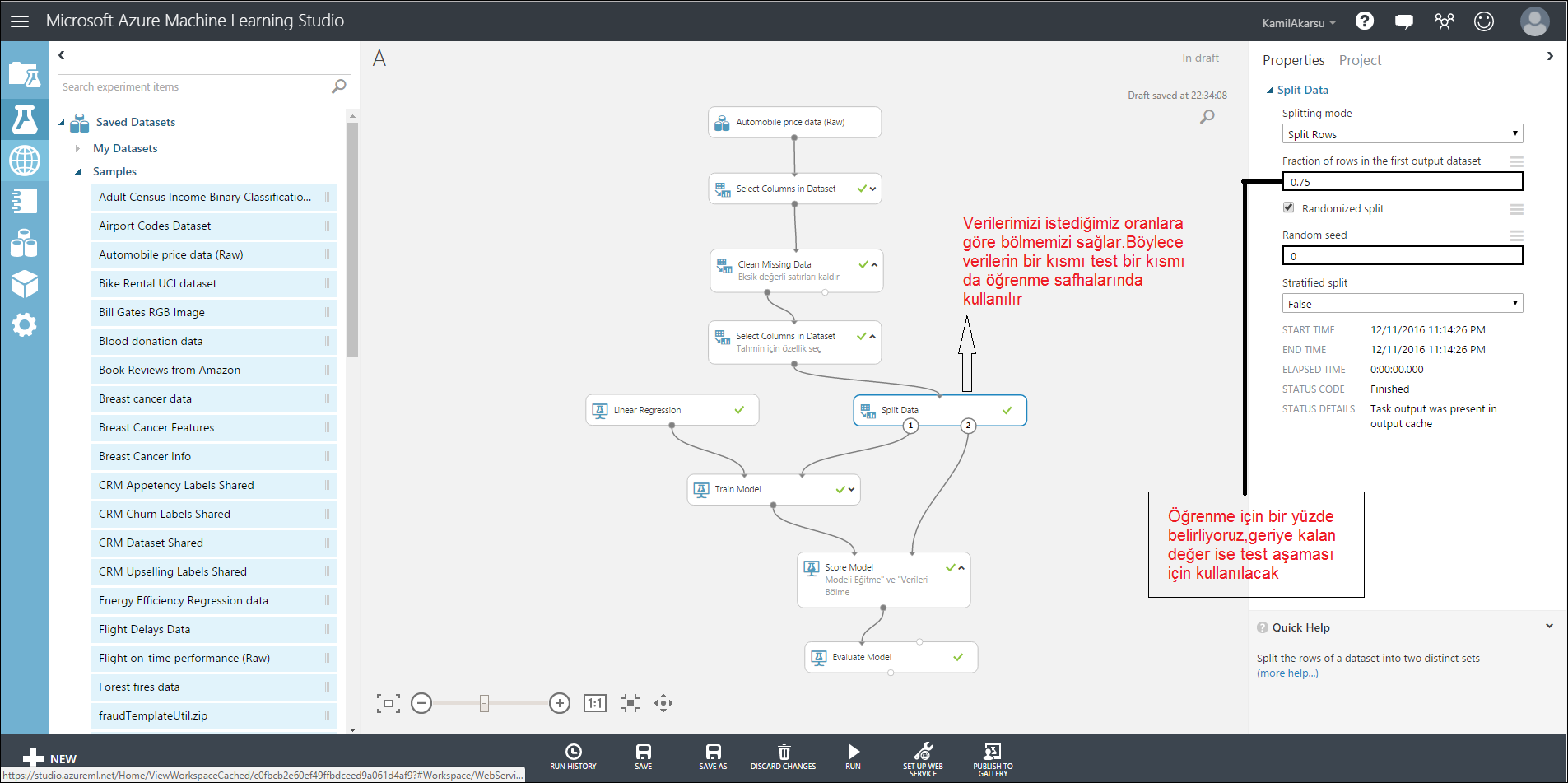


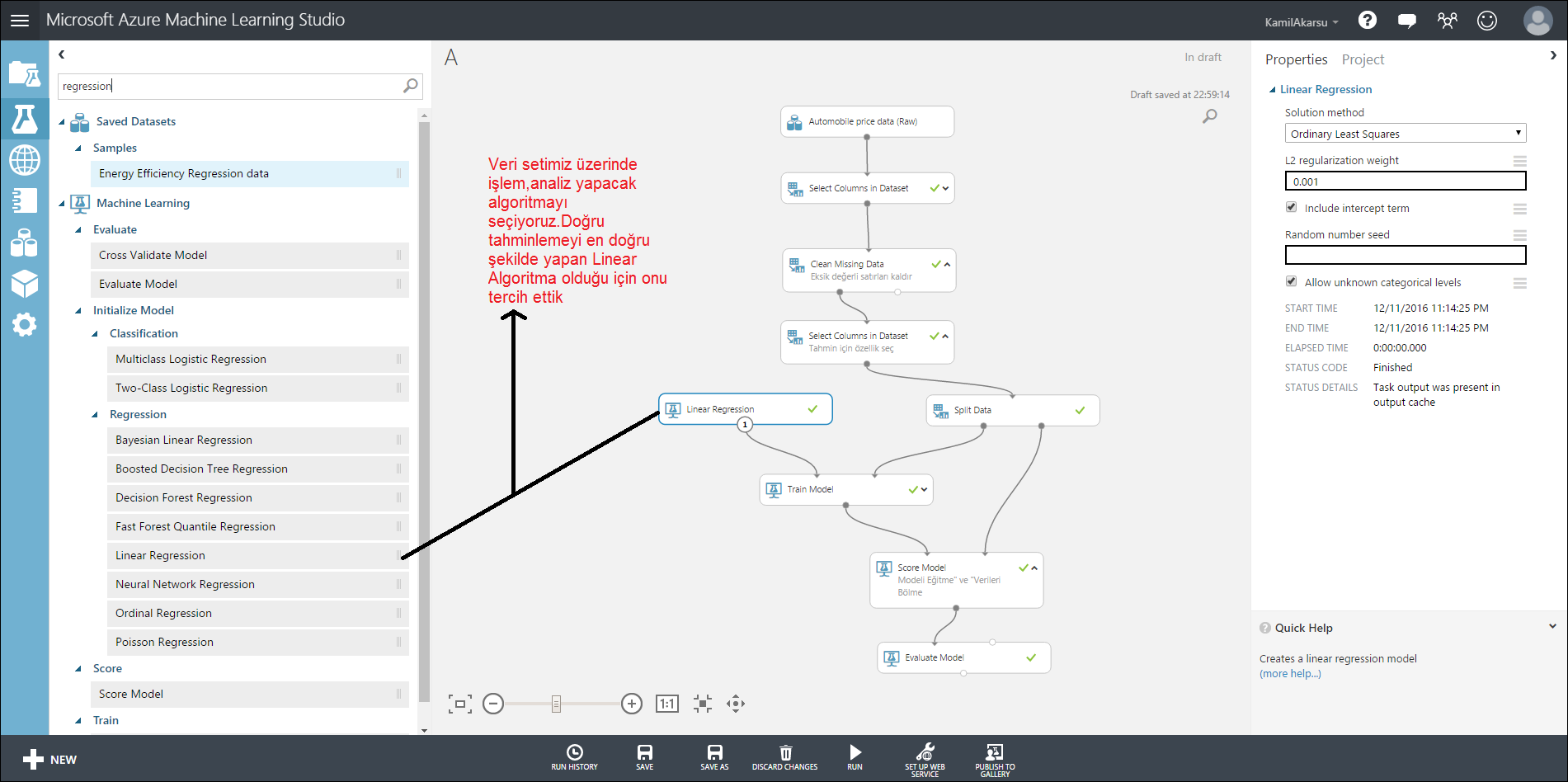


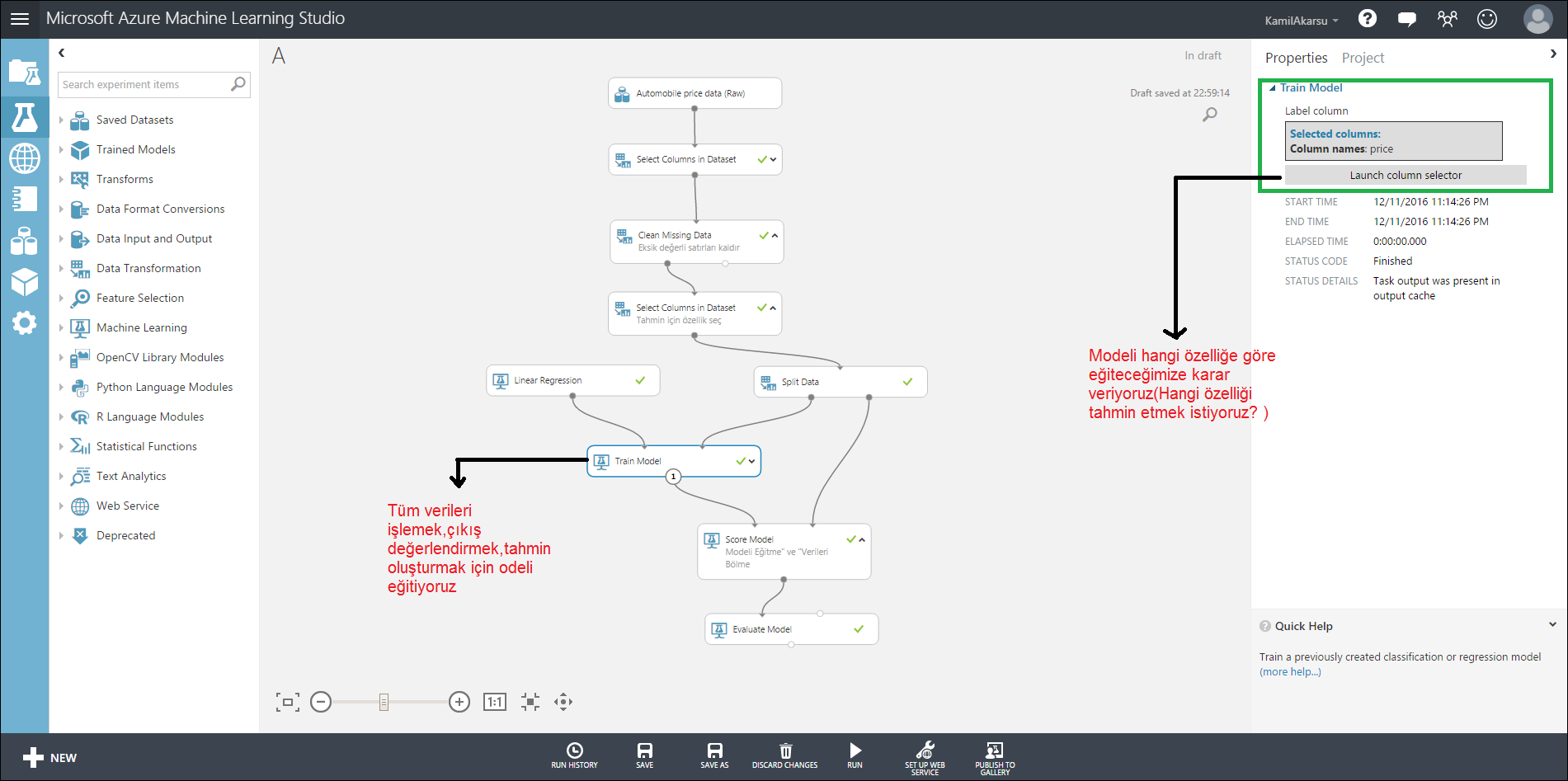


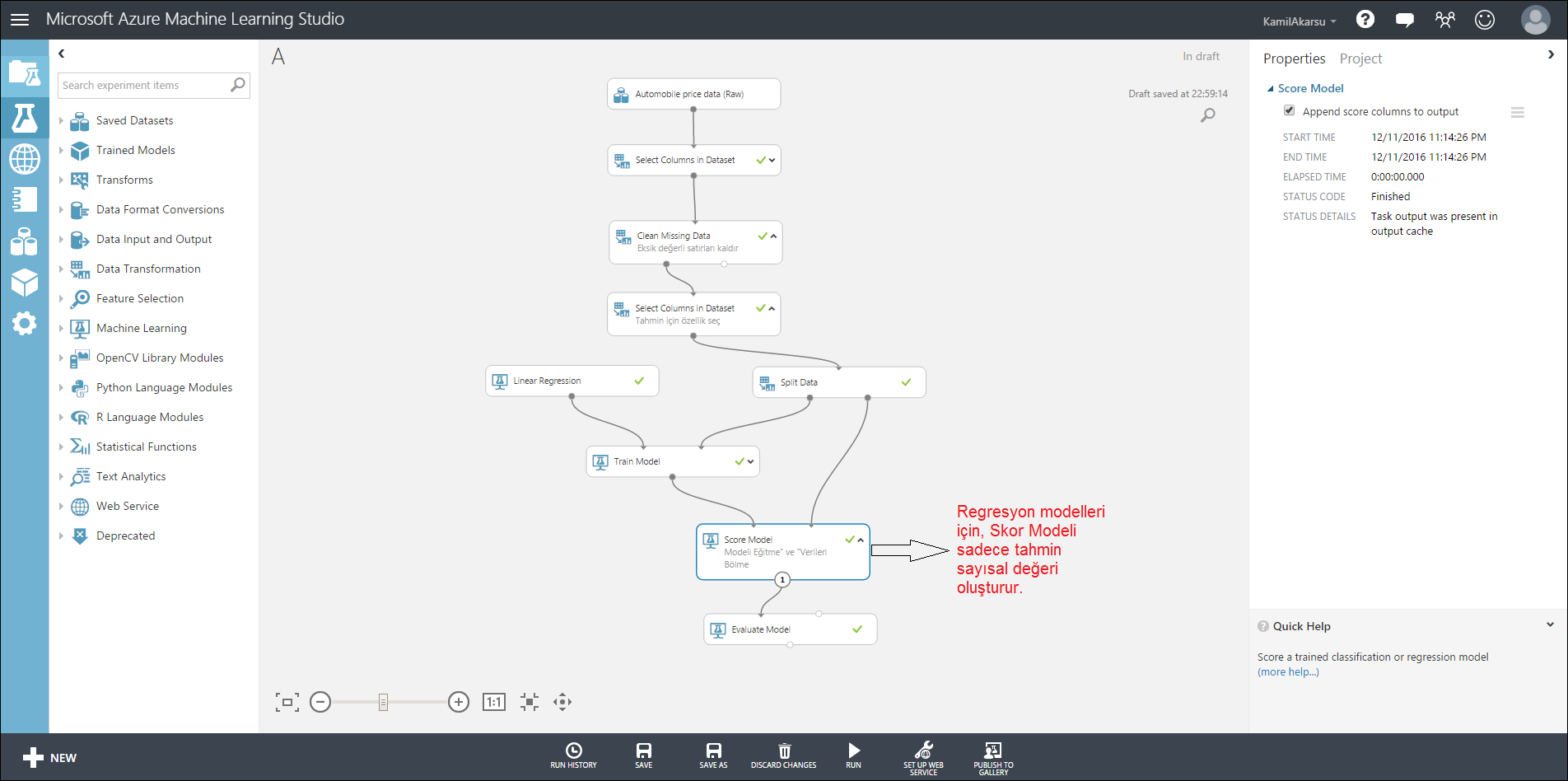


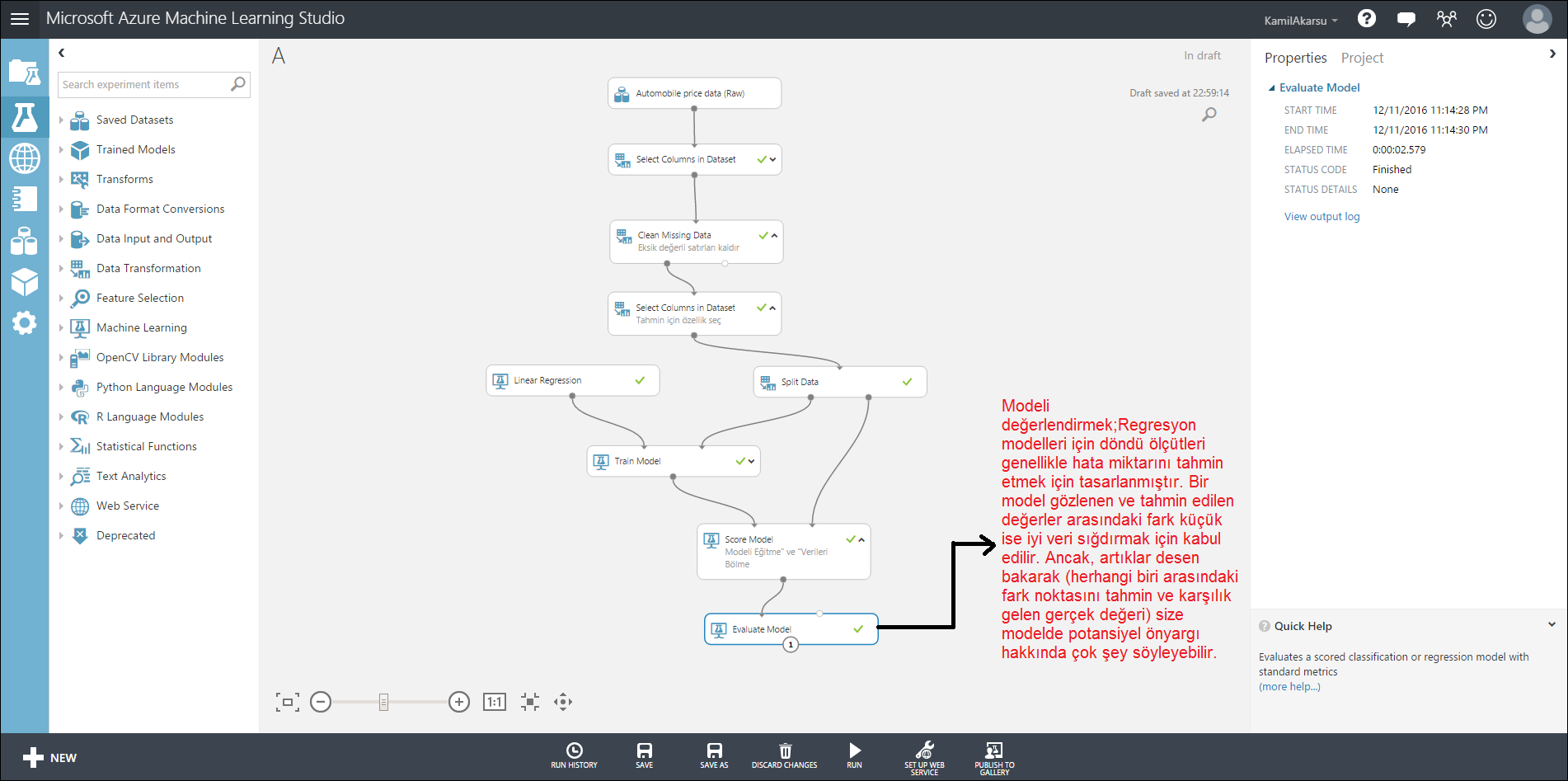




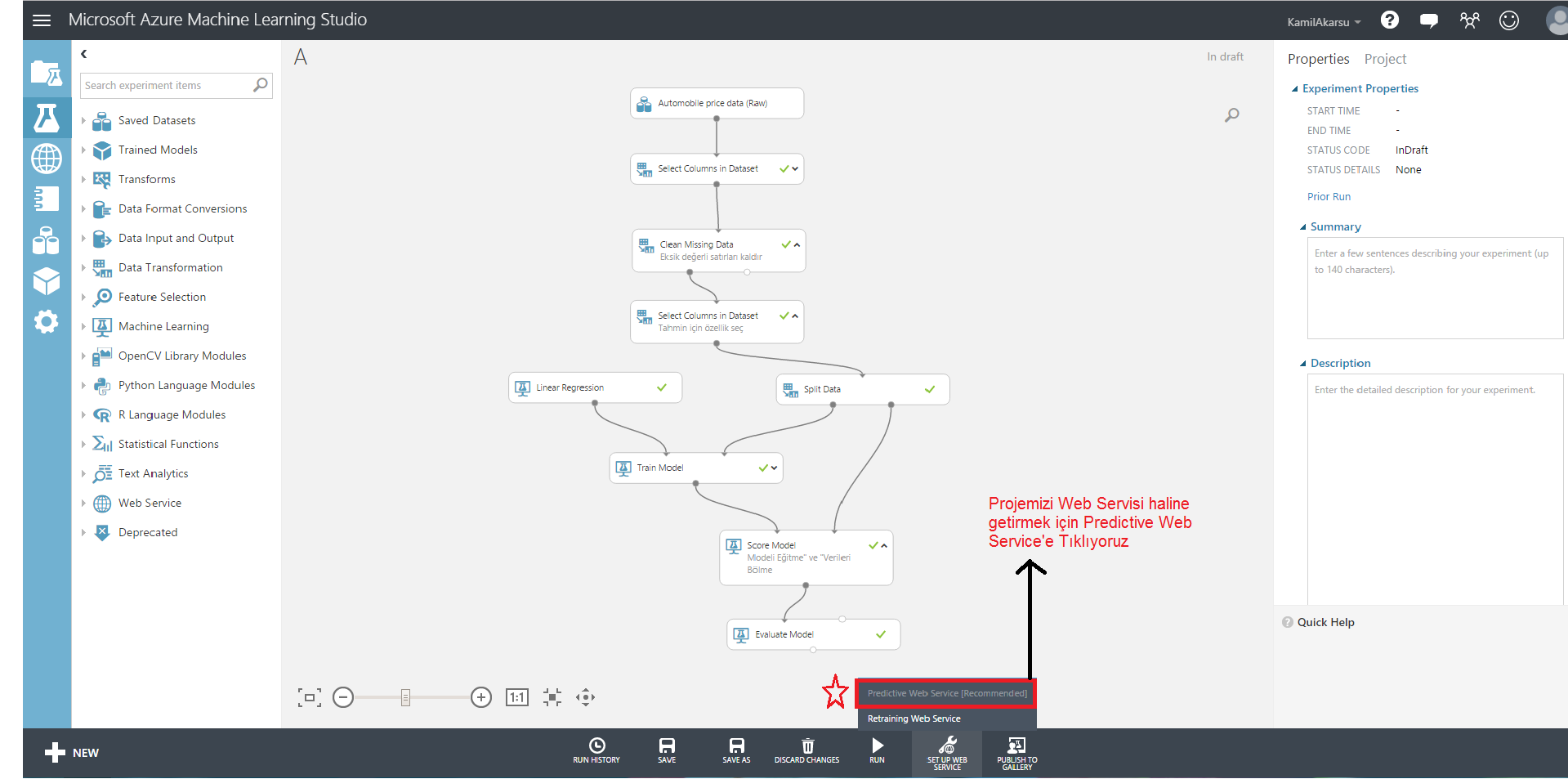


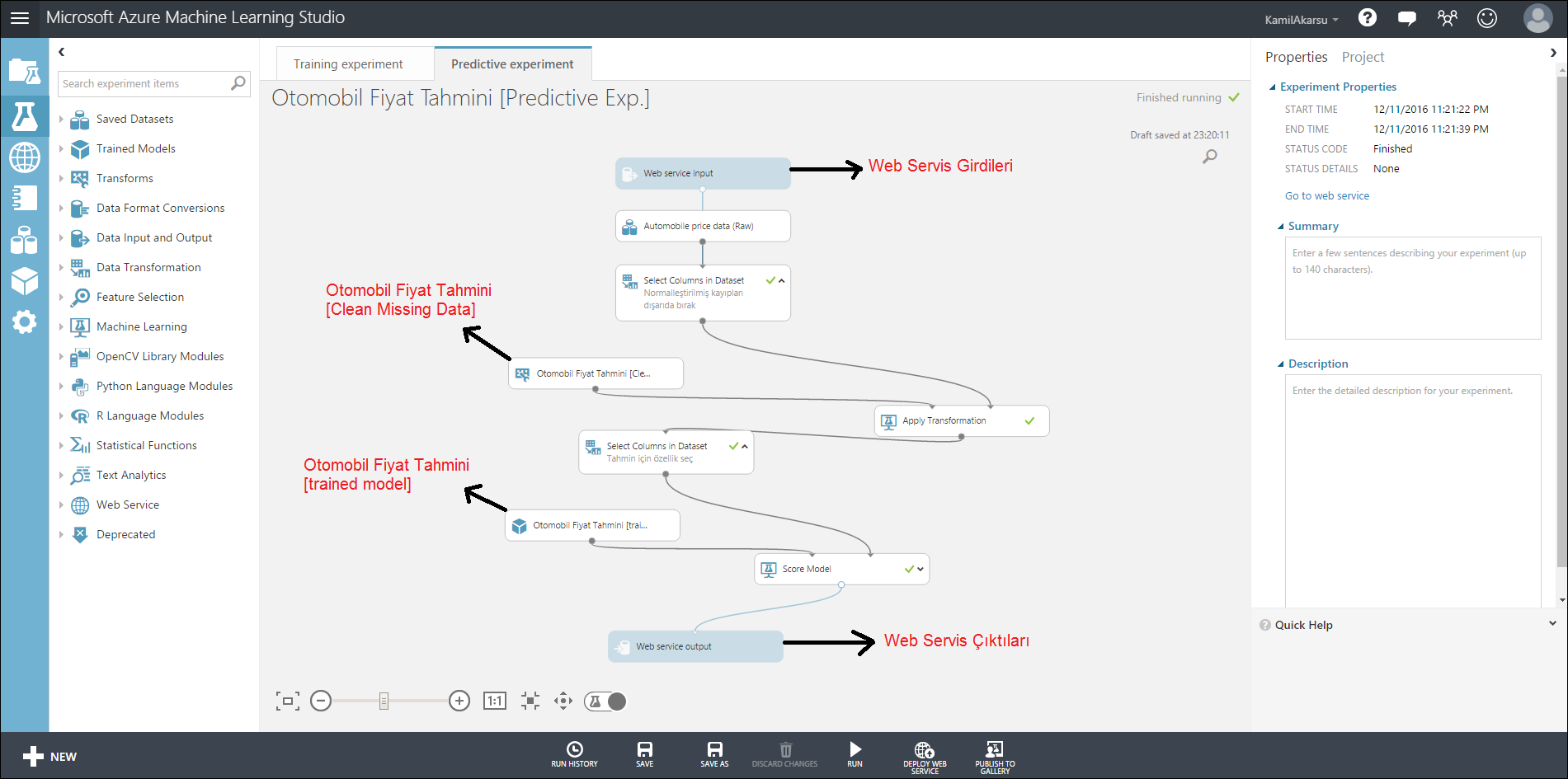


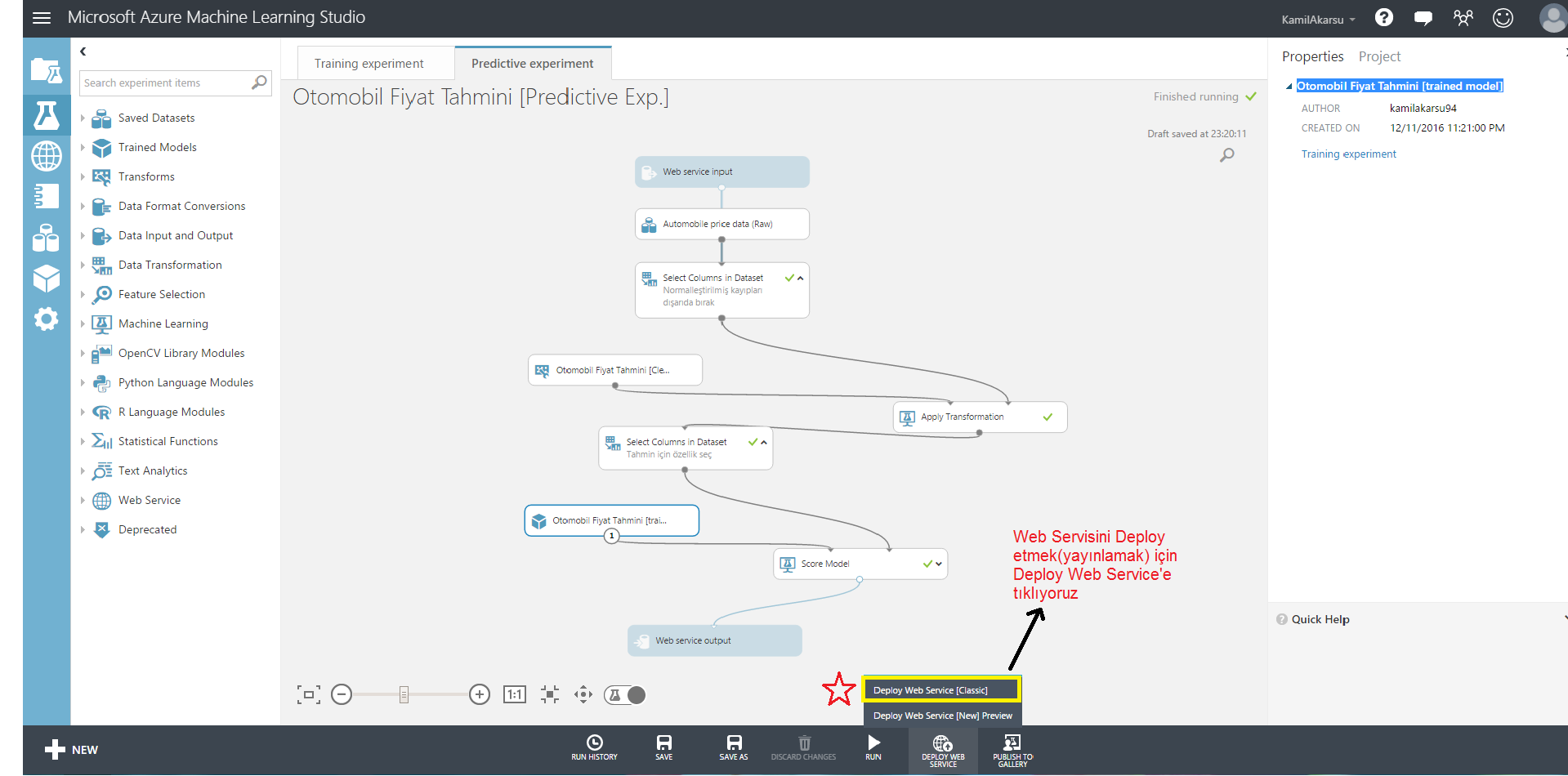


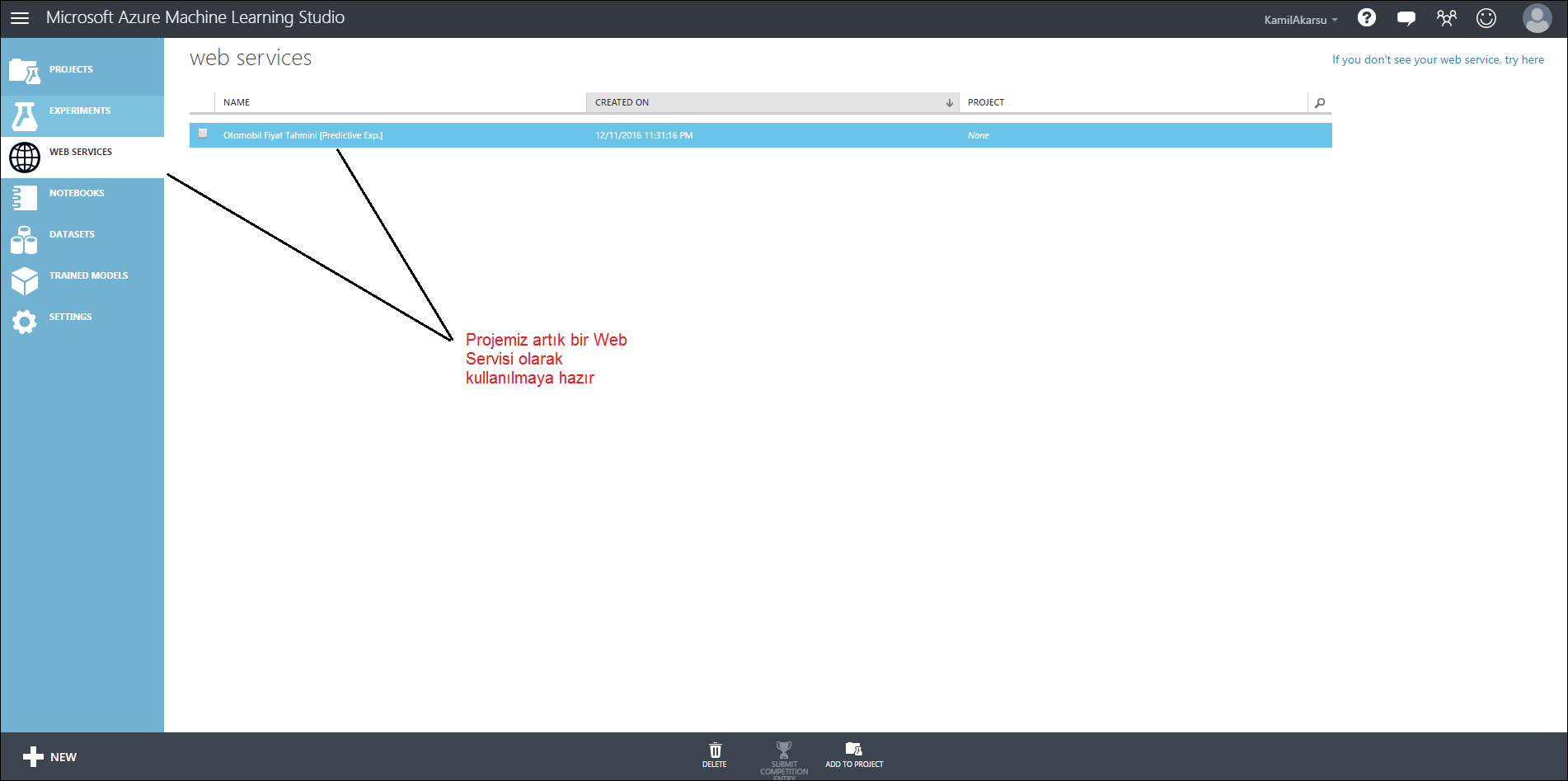


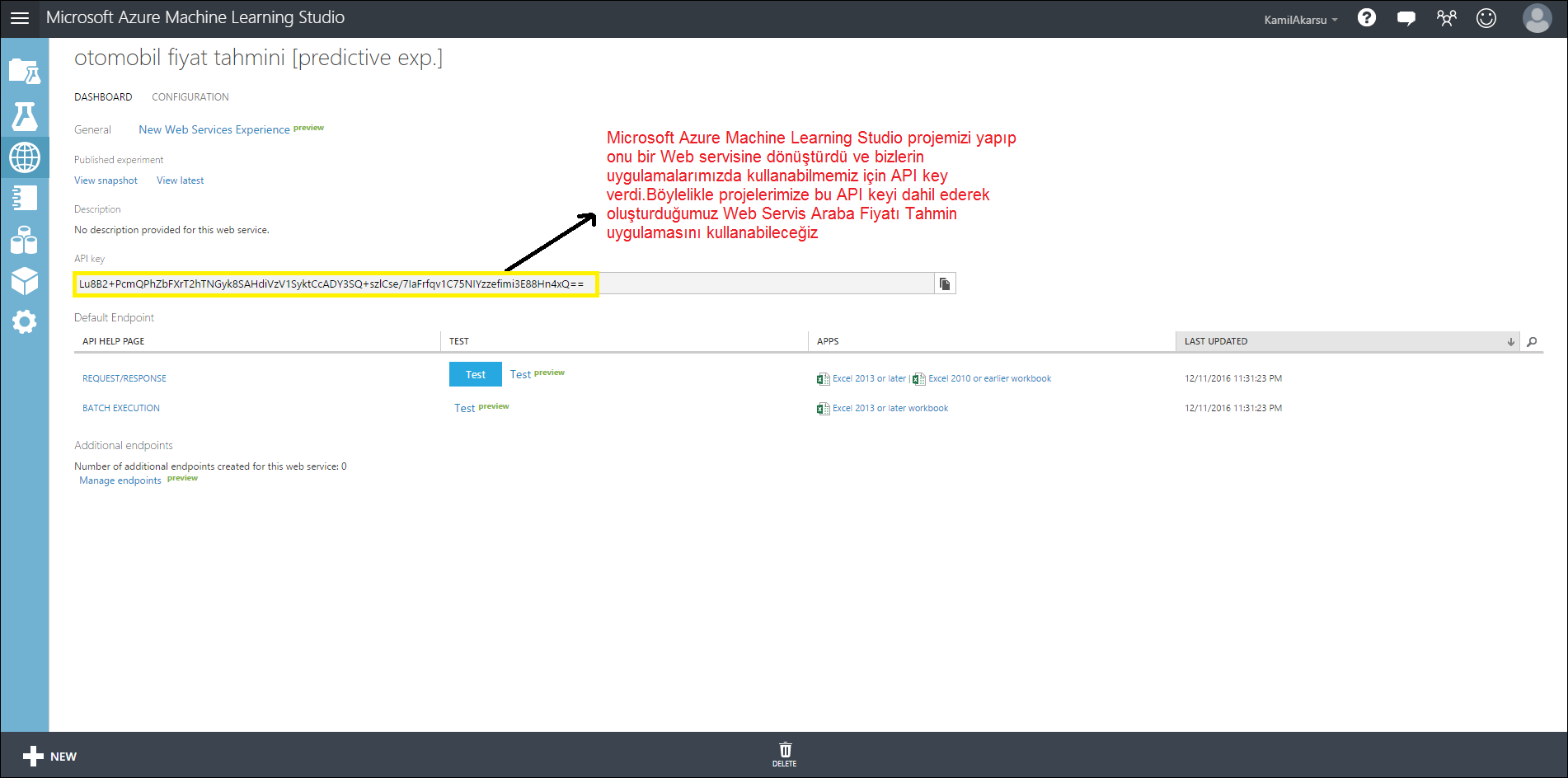
5. Oluşturulan Modeli Web Servisi Olarak Yayınlama



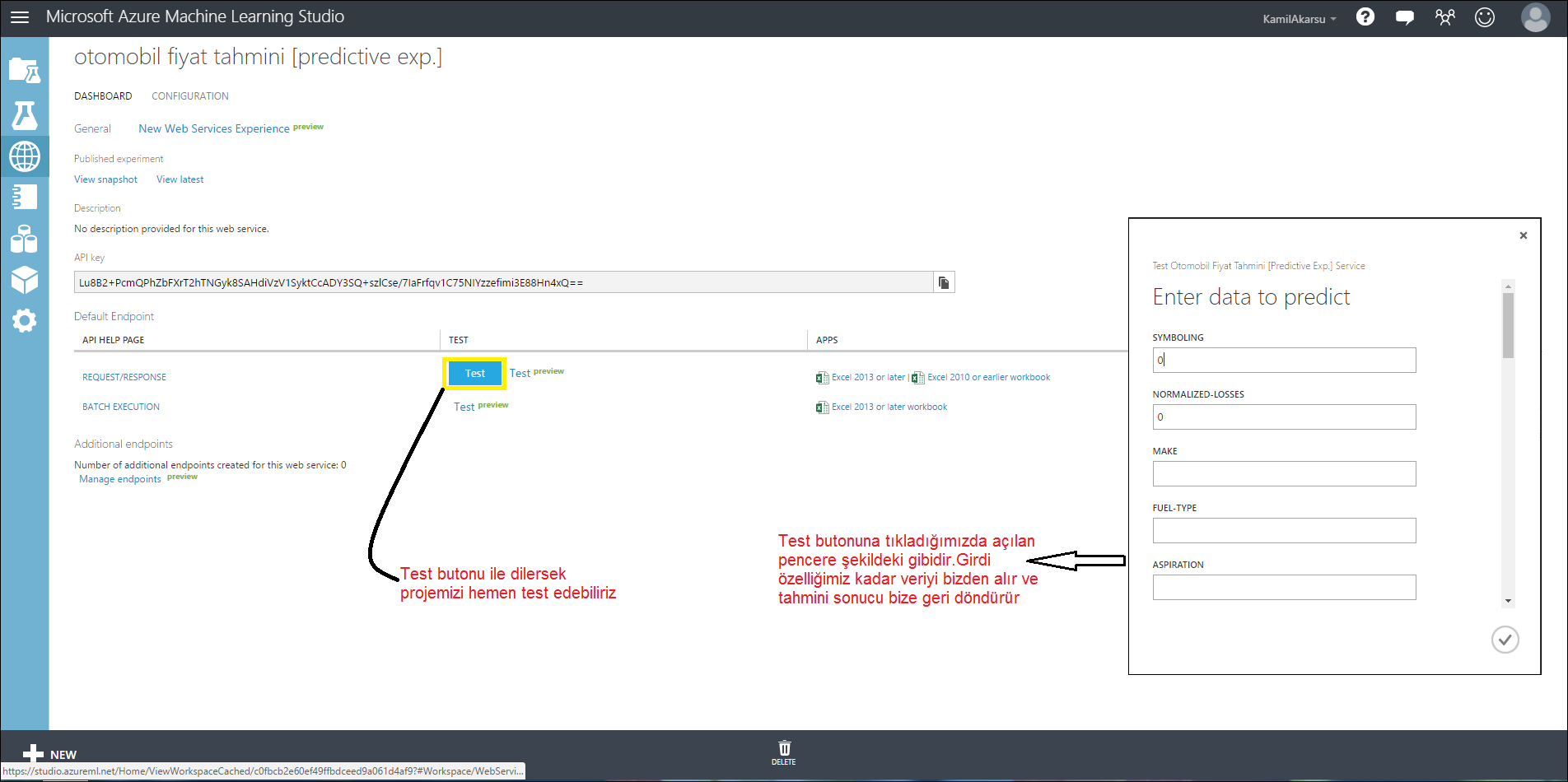


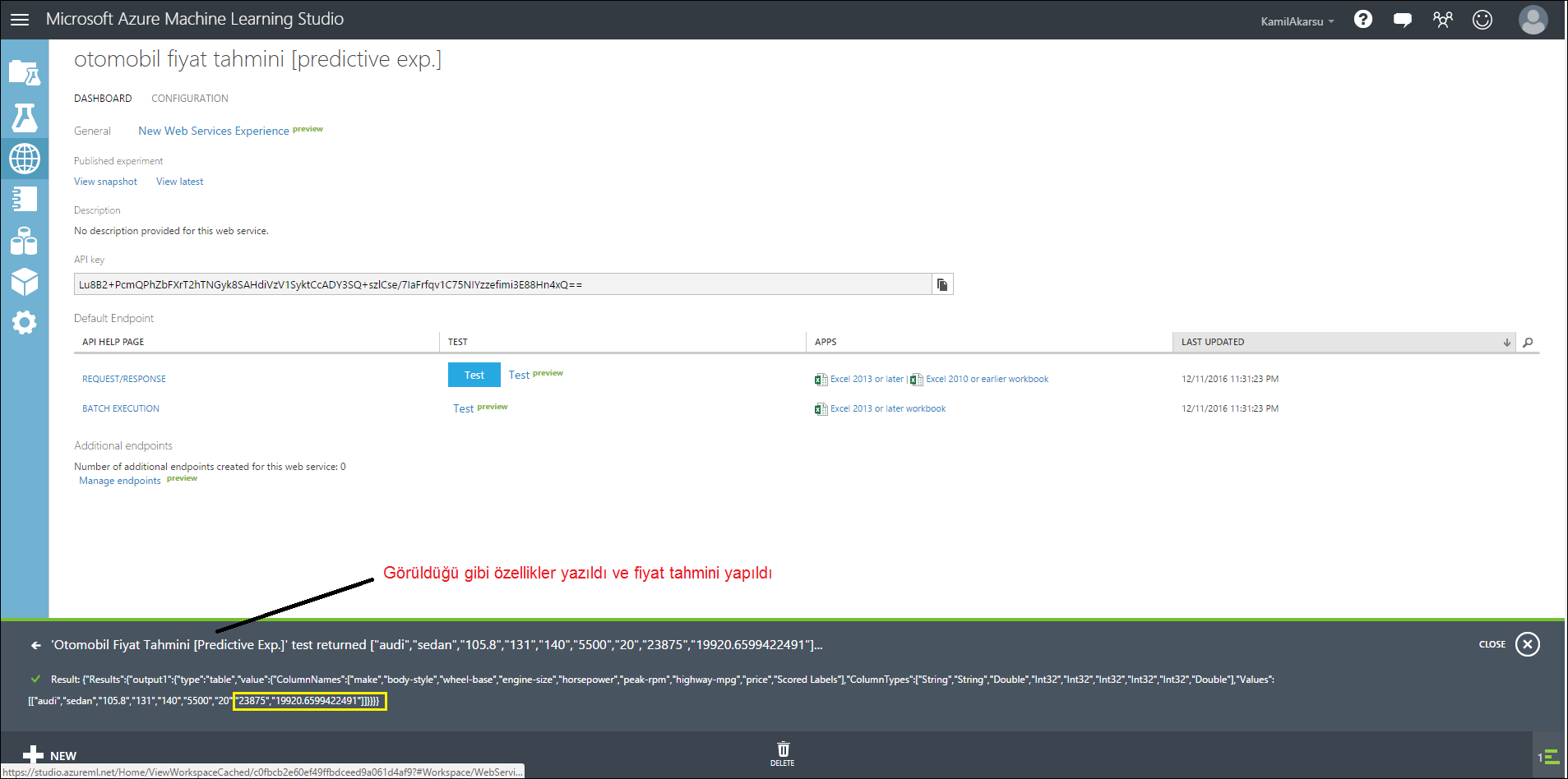






6. Projenin Microsoft Azure Machine Learning Studio Ortamında Test Edilmesi





7. Programın Kaynak Kodları

using System;

using System.Collections.Generic;

using System.IO;

using System.Net.Http;

using System.Net.Http.Formatting;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace CallRequestResponseService

{

public class StringTable//Request ile gönderilecek olan kolonlar ve değerleri için

{

public string[] ColumnNames { get; set; }

public string[,] Values { get; set; }

}

class Program

{

static void Main(string[] args)

{

InvokeRequestResponseService().Wait();//Async bağlantılarda wait kullanılır

}

static async Task InvokeRequestResponseService()

{

string temp="";

using (var client = new HttpClient())

{

var scoreRequest = new

{

Inputs = new Dictionary<string, StringTable> ()

{

{

"input1",

new StringTable()

{

// "1","158","audi","gas","turbo","four","sedan","fwd","front","105.80","192.70","71.40","55.90","3086","ohc","five","131","mpfi","3.13","3.40","8.30","140","5500","17","20","23875"

// "1","154","alfa-romero","gas","std","two","hatchback","rwd","front","94.50","171.20","65.50","52.40","2823","ohcv","six","152","mpfi","2.68","3.47","9.00","154","5000","19","26","16500"

// "0","192","bmw","gas","std","four","sedan","rwd","front","101.20","176.80","64.80","54.30","2395","ohc","four","108","mpfi","3.50","2.80","8.80","101","5800","23","29","16925"

// "2","103","volvo","gas","turbo","four","sedan","rwd","front","104.30","188.80","67.20","56.20","3045","ohc","four","130","mpfi","3.62","3.15","7.50","162","5100","17","22","18420"

ColumnNames = new string[] {"symboling", "normalized-losses", "make", "fuel-type", "aspiration", "num-of-doors", "body-style", "drive-wheels", "engine-location", "wheel-base", "length", "width", "height", "curb-weight", "engine-type", "num-of-cylinders", "engine-size", "fuel-system", "bore", "stroke", "compression-ratio", "horsepower", "peak-rpm", "city-mpg", "highway-mpg", "price"},

Values = new string[,] { {"2","103","volvo","gas","turbo","four","sedan","rwd","front","104.30","188.80","67.20","56.20","3045","ohc","four","130","mpfi","3.62","3.15","7.50","162","5100","17","22","18420"} }

}

},

},

GlobalParameters = new Dictionary<string, string>() {}

};

const string apiKey = "Lu8B2+PcmQPhZbFXrT2hTNGyk8SAHdiVzV1SyktCcADY3SQ+szlCse/7IaFrfqv1C75NIYzzefimi3E88Hn4xQ=="; // Azure'un bize verdiği API KEY ile bağlanıcaz

client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue( "Bearer", apiKey);

client.BaseAddress = new Uri("https://ussouthcentral.services.azureml.net/workspaces/c0fbcb2e60ef49ffbdceed9a061d4af9/services/d6aaa1bfe31c4258b7c8bbb580575105/execute?api-version=2.0&details=true");

HttpResponseMessage response = await client.PostAsJsonAsync("", scoreRequest);

if (response.IsSuccessStatusCode)

{

string result = await response.Content.ReadAsStringAsync();

string metin =result ;

string[] parcalar = metin.Split('[',']',':','<','>','{','}');

foreach (string parca in parcalar)

{

temp += parca + Environment.NewLine;

}

Console.WriteLine(temp);

//MessageBox.Show(temp);

}

else

{

Console.WriteLine(string.Format("The request failed with status code: {0}", response.StatusCode));

Console.WriteLine(response.Headers.ToString());

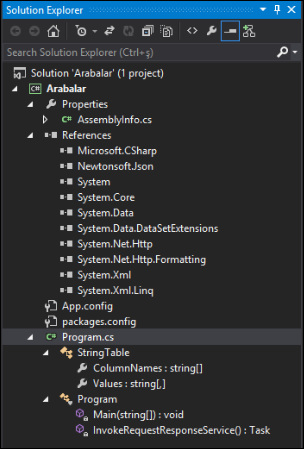
string responseContent = await response.Content.ReadAsStringAsync();

Console.WriteLine(responseContent);

}

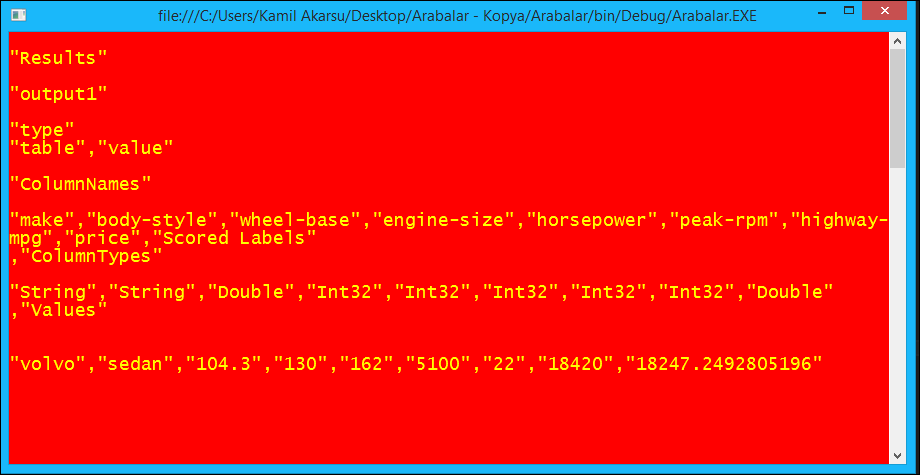
System.Threading.Thread.Sleep(30008800);

}}}}

8. Proje de kullanılan referanslar ve kütüphaneler

9. Program Ekran Çıktıları

9.1. Ekran Çıktısı 1



9.2. Ekran Çıktısı 2



9.3. Ekran Çıktısı 3



9.4. Ekran Çıktısı 4

