PİYASA DEĞER TAHMİN ALGORİTMASI DÖNEM SONU RAPORU

1.GİRİŞ

Bil 401 dersi projesi kapsamında, futbol sektöründe oyuncu piyasa değerlerinin oyuncu öznitelikleri doğrultusunda tahmin edilmesi adına büyük veriyi işleyerek yapay öğrenme modelinin eğitilmesi işlemini hızlandırmak hedeflenmiştir. Proje süresince veriyi işlemek, görselleştirmek ve modeli eğitmek adına kullanılabilecek teknolojiler araştırılmış, ilgili teknolojilerinin kullanımı hakkında deneyim elde edilmiştir. Raporda sırası ile kullanılan teknolojiler ,veri seti incelenmesi ve model eğitimi hakkında detaylı bilgilere alt başlıklar altında yer verilmiştir.

2.KULLANILAN TEKNOLOJİLER

Proje python programlama dili ile Jupyter Notbook üzerinde geliştirilmiştir. Projede, 9.42 KB büyüklüğündeki bir veri seti kullanılmıştır. Büyük veri kullanılmamasına karşın kullanılan teknolojiler büyük veri işlemeye yönelik olarak araştırılmış ve seçilmiştir. Proje süresince pyspark, pandas, handyspark, numpy ve matplotlib kütüphanelerinden yararlanılmıştır. Model eğitimi için pyspark içerisinde bulunan LinearRegression ve RandomForestRegressor modelleri kullanılmıştır. İlgili modelleri kullanabilmek için özniteliklerin tek bir vektör özniteliği altında toplanması gerekmektedir. Bunun için de pyspark içerisinde bulunan VectorAssembler teknolojisinden yararlanılmıştır. Yine model eğitimi için sayısal olmayan özniteliklerin sayısallaştırılması işlemi pyspark içerisinde bulunan StringIndexer teknolojisi yardımı ile gerçekleştirilmiştir. Projede kullanılan teknolojiler hakkında teknik ve detaylı bilgilere aşağıda yer verilmiştir.

- **Jupyter Notebook:** Jupyter Notebook, çeşitli programlama dilleri için etkileşimli bir ortam sağlayan açık kaynak kodlu bir programdır. Notebook kelimesinden de anlaşılacağı üzere kodlarınızın bir not defterinde tutulması gibi bir hizmet sunar. Daha basit olarak ele alırsak word belgesi içerisinde python kodlarının çalıştırılmasını ele alabiliriz. Kimisi ödevleri için not defteri olarak kullanır, kimileri kodlarını test edebileceği ve aynı zamanda onları not alabileceği bir ortama aktarmak için kullanır, kimisi de üst satıda toplama işlemi yaptırırken alt satırda makine öğrenimi çalıştıran bir kod yazmak için kullanır.

- **Handyspark:** HandySpark, özellikle görselleştirme yetenekleri de dahil olmak üzere keşifsel veri analizi söz konusu olduğunda, PySpark kullanıcı deneyimini geliştirmek için tasarlanmış bir pakettir. Sütunlar için veri getirmeyi ve istatistikleri hesaplamayı kolaylaştırır ve pandas nesnelerini hemen döndürür.

- **Matplotlib:** Matplotlib; veri görselleştirmesinde kullandığımız temel python kütüphanesidir. 2 ve 3 boyutlu çizimler yapmamızı sağlar. Matplotlib genelde 2 boyutlu çizimlerde kullanılır.

- **Numpy:** NumPy, Python programlama dili için büyük, çok boyutlu dizileri ve matrisleri destekleyen, bu diziler üzerinde çalışacak üst düzey matematiksel işlevler ekleyen bir kitaplıktır NumPy'nin temel işlevi, n-boyutlu dizi veri yapısı için "ndarray"idir. Bu diziler, bellekte adım adım görünümlerdir. Python'un yerleşik liste veri yapısının aksine, bu diziler homojen olarak yazılmaktadır. Tek bir dizinin tüm elemanları aynı tipte olmalıdır.

- **Pandas:** Pandas, veri işlemesi ve analizi için Python programlama dilinde yazılmış olan bir yazılım kütüphanesidir. Bu kütüphane temel olarak zaman etiketli serileri ve sayısal tabloları işlemek için bir veri yapısı oluşturur ve bu şekilde çeşitli işlemler bu veri yapısı üzerinde gerçekleştirilebilir olur.

  Kütüphane özellikleri:

- İndeksli DataFrame (veri iskeleti) objeleri ile veri işlemesi yapabilmek.
- Veri sıralama ve bütünleşik kayıp veri senaryolarına karşı esnek imkanlar sunması.
- Veri setlerinin tekrar boyutlandırılması veya döndürülmesi.
- Etiket bazlı dilimleme, özel indeksleme ve büyük veri setlerini ayrıştırma özelliği.
- Veri iskeletine sütun ekleme veya var olan sütunu çıkarma.
- Veri gruplama özelliği ile ayırma-uygulama-birleştirme uygulamalarının yapılabilmesi.
- Veri setlerinin birleştirilmesi ve birbirine eklenmesi.
- Hiyerarşik eksenleri indeksleme özelliğiyle birlikte çok boyutlu veriden, daha az boyutlu veri elde edilebilmesi.
- Veri filtrelemesi yapabilmek.

- **LinearRegression**: Linear Regresyon analizi, bir değişkenin değerini başka bir değişkenin değerine göre tahmin etmek için kullanılır. Tahmin etmek istediğiniz değişken, bağımlı değişken olarak adlandırılır. Diğer değişkenin değerini tahmin etmek için kullandığınız değişken ise bağımsız değişken olarak adlandırılır. Bu analiz biçimi, bağımlı değişkenin değerini en iyi öngören bir ya da daha fazla bağımsız değişkeni kullanarak doğrusal denklemin katsayılarını tahmin eder. Doğrusal regresyon, öngörülen ve gerçek çıkış değerleri arasındaki uyumsuzlukları en aza indiren düz bir çizgi ya da yüzeye yerleşir. Bir çift eşleştirilmiş veri kümesi için en uygun satırı keşfetmek üzere "en küçük kareler" yöntemini kullanan basit doğrusal regresyon hesaplayıcılar vardır. Daha sonra, Y'den (bağımsız değişken) X'in (bağımlı değişken) değerini tahmin edersiniz.

- **RandomForestRegressor**: Random Forest algoritması; birden çok karar ağacı üzerinden her bir karar ağacını farklı bir gözlem örneği üzerinde eğiterek çeşitli modeller üretip, sınıflandırma oluşturmanızı sağlamaktadır. Kullanım kolaylığı ve esnekliği; hem sınıflandırma hem de regresyon problemlerini ele aldığı için benimsenmesini ve kullanımının yaygınlaşmasını hızlandırdı. Algoritmanın en iyi

yönü ise beğenilen nokta ise; veri kümeniz üzerinde çeşitli modellerin oluşturulması ile kümenizi yeniden ve daha derin keşfetme imkanı sunmasıdır.

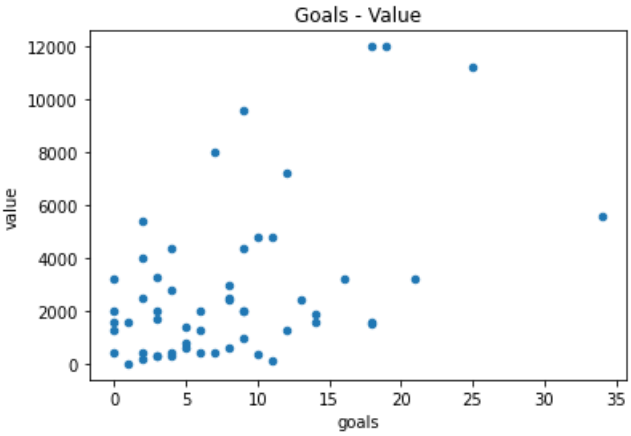
- **VectorAssembler:** Birden çok sütunu bir vektör sütunuyla birleştiren bir fonksiyondur.

- **StringIndexer:** Makine Öğrenimi algoritmasının sütunu kategorik değişken olarak tanımlamasını istiyorsanız veya kategorik bağlamı koruyarak metinsel verileri sayısal verilere dönüştürmek istiyorsanız bunu kullanılır.
  örneğin günleri (Pazartesi, Salı...) sayısal gösterime dönüştürmek.

3.VERİ SETİNİN İNCELENMESİ

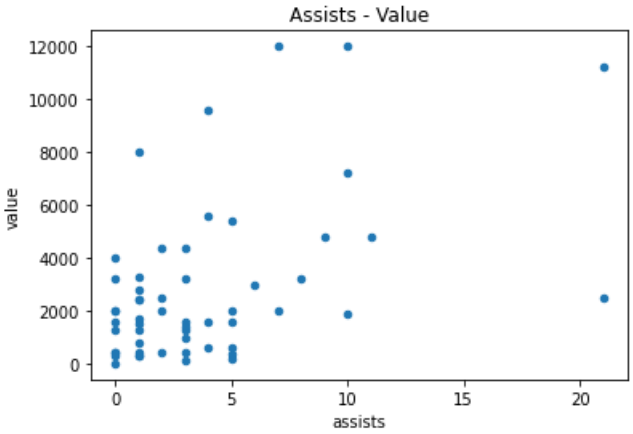
Veri setinde League, player, Substitution, Match, Shots, OnTarget, Shots Per Avg Match, On Target Per Avg Match, value, nationality, position, squad, goals, goals_per_shot, passes_completed, assisted_shots, passes_blocked, touches, assists, games_starts, minutes ve squad satırları bulunmaktadır. Görsel 1'de veri setindeki satırlar hakkında daha detaylı bilgiler yer almaktadır. Proje Avrupa'nin beş büyük ligine odaklandığından bu liglerden biri olmayan MLS ligine ait sütün veri setinden atılmıştır. Veri setindeki indekslerin gösterildiği sütun da (Team) eğitime katkısı olmacayacağından ötürü veri setinden çıkarılmıştır. Belirli sütunlarda League – squad öznitelikleri arasında mantıksal bir hata olduğu, veri setindeki bazı takımların farklı ülkelere ait liglerle aynı sütunda yer aldığı saptanmıştır. Sorunu çözmek amacı ile ilk olarak veri setinden hatalı değerler içeren League özniteliği atılmıştır. Daha sonrasında veri setinin referans aldığı futbol sezonu baz alınarak takımlar ve takımların bulunduğu ligleri içeren ayrı bir tablo oluşturulmuştur. Son olarak, iki veri setine squad özniteliği baz alınarak join işlemi uygulanmıştır. Join işlemi sonrasında duplicate satırlar atılmıştır.

Veri seti incelenirken mantıksal ve yapısal hataların tespit edilmesi ve düzeltilmesi amaçlanmıştır. Veri setinde eksik veri bulunmamıştır. Veri setinde en çok oyuncusu bulunan ligin 16 oyuncu ile Serie A olduğu tespit edilmiştir. Proje kapsamında veriyi anlaşılabilir kılması, veri işleme aşamasına yardımcı olması amacı ile veri setindeki belirli öznitelikler ve birbirleri ile ilişkileri görselleştirilmetye çalışılmıştır. Grafiklere ve yorumlarına aşağıda yer verilmiştir.



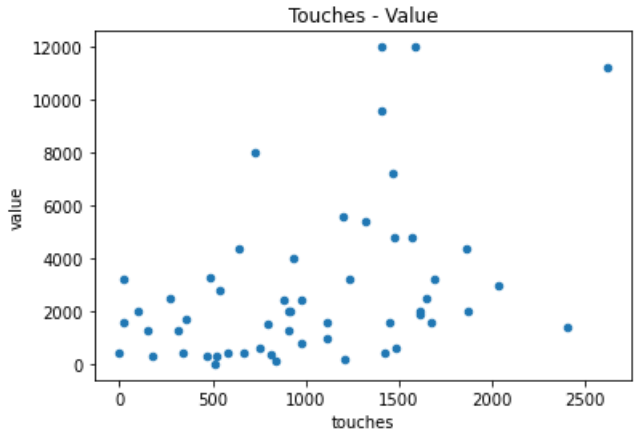
Grafik 1

Grafik 1, veri setindeki oyuncuların sezon içerisindeki gol sayıları (x) ve piyasa değerlerini (y) göstermektedir. Bu noktada oyuncuların gol sayıları ile piyasa değerleri arasında doğrudan bir ilişki olmadığı gözlemlenmiştir.



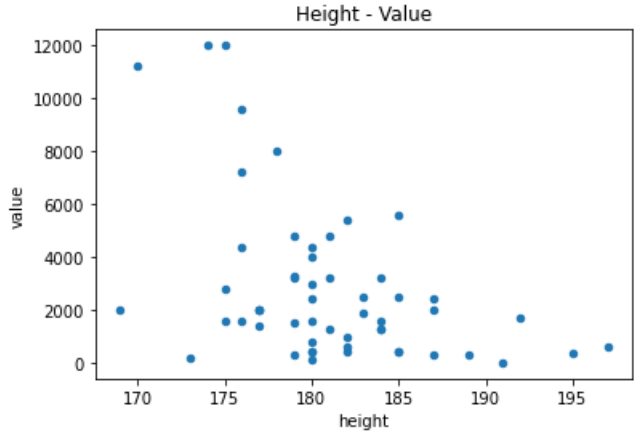
Grafik 2

Grafik 2, veri setindeki oyuncuların sezon içerisindeki asist sayıları (x) ve piyasa değerlerini (y) göstermektedir. İlgili grafik ile Grafik 1 üzerinde benzer sonuçlara varılmıştır. Oyuncuların asist sayıları ile piyasa değerleri arasında doğrudan bir ilişki olmadığı gözlemlenmiştir.



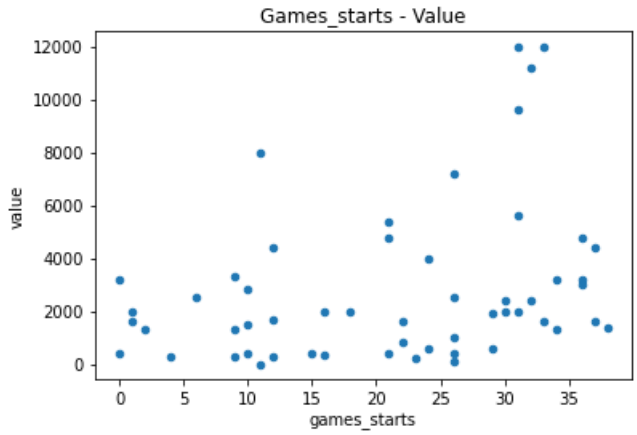
Grafik 3

Grafik 3, veri setindeki oyuncuların sezon içerisinde topa doknuş sayıları (x) ve piyasa değerlerini (y) göstermektedir. Grafik 3, beklenene yakın bir sonucu göstermektedir. Topa az doknunan oyuncuların piyasa değerlerinin yüksek olmadığı görülmüştür. Model eğitimi süresince bu özniteliğin önemli olacağı ön görülmektedir.



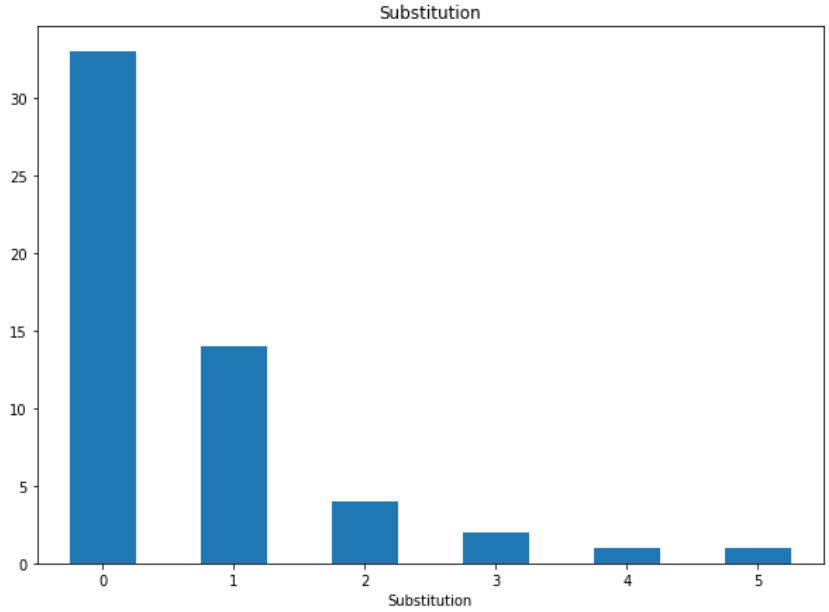
Grafik 4

Grafik 4, veri setindeki oyuncuların boyları (x) ve piyasa değerlerini (y) göstermektedir. Grafiğe bakıldığında kısa boylu oyuncuların daha yüksek değere sahip olduğu görülmüştür. Ancak iki öznitelik arasındaki bu ilişkinin tesadüfi olduğu düşünülmektedir. Buna ek olarak veri seti de yeterince büyük olmadığından bu grafikten kesin sonuçlar çıkarmak mümkün değildir.

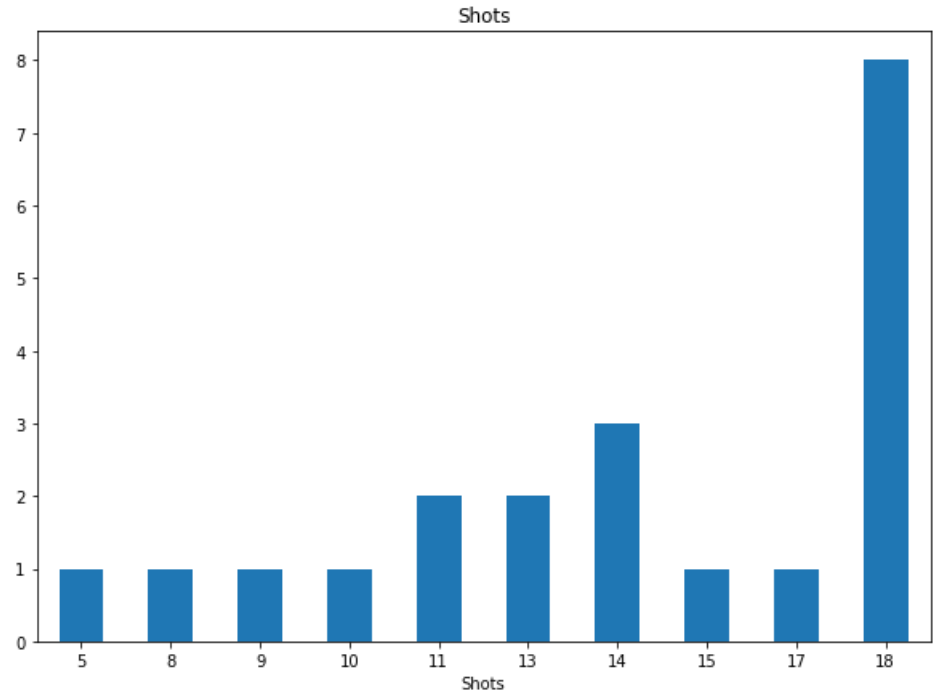


Grafik 5

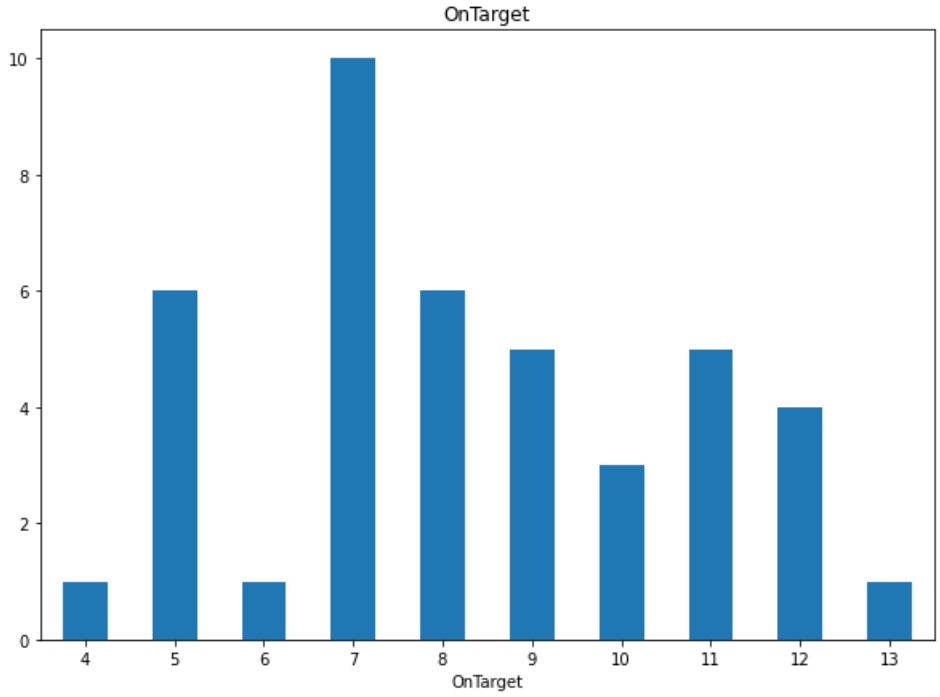
Grafik 5, veri setindeki oyuncuların ilk on birde başladıkları toplam maç sayıları (x) ve piyasa değerlerini (y) göstermektedir. Grafik 5 bazı aykırı değerler harici beklenilen bir sonucu bize vermektedir. Oynamayan bir oyuncunun piyasa değerinin artmasını beklemek çok da mantıklı değildir. Veri setindeki en yüksek piyasa değerine sahip oyuncu kümesine bakıldığında da ilk on birde daha fazla maça başladıkları görülmektedir.



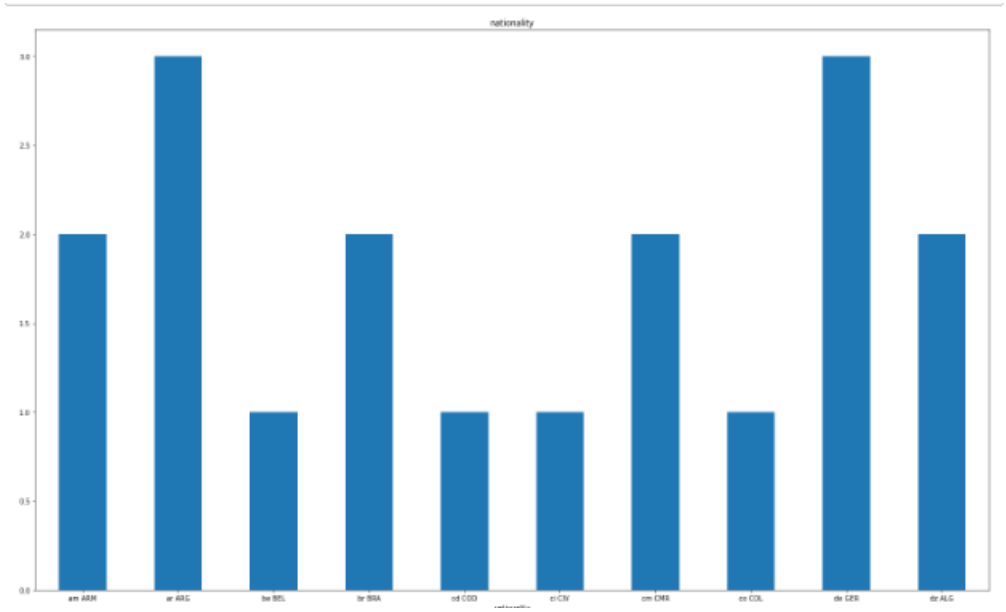
Grafik 6, veri setindeki oyuncuların oyun esnasında değiştikleri sayıları vermektedir. Oyundan alınma oyuncunun piyasa değeri hakkında belirleyici bir etken göstermeyebilir ama çokca maçta oyundan alınmış bir oyuncunun piyasasının artmaması da beklenir.



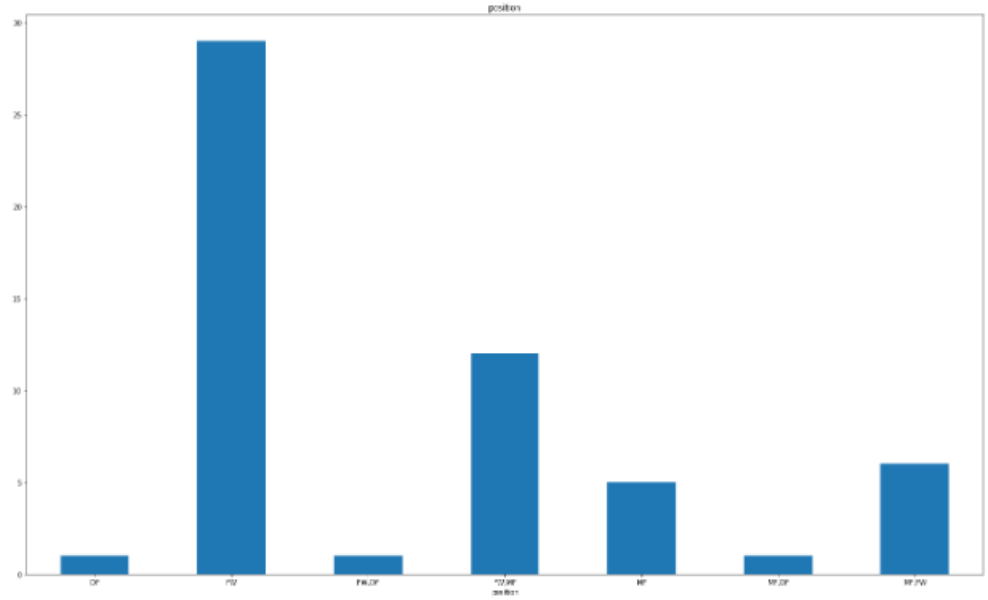
Grafik 7, veri setindeki oyuncuların attıkları gollerin kaç şutta sonuca ulaştığını bize verir. Shots ve gol verileri arasında ki orantı bize oyuncunun deneme sayılarını ve takımda ki potansiyel pozisyonların kaç tanesini gole çevirdiğni bize verir.



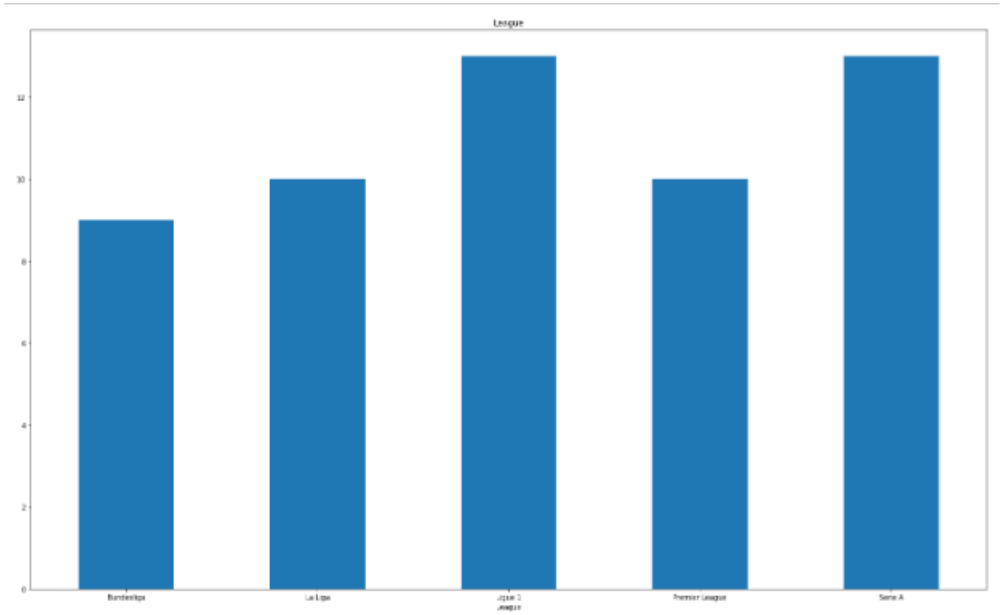
Grafik 8, veri setindeki oyuncuların çektikleri şutların kaçının hedefi bulduğunu bize gösterir. Bu veri de aslında gol yapma yüzdesiyle doğrudan bağlantılıdır. Hedefi bulan futbolcuların piyasasının artmasını bekleriz



Grafik 9, veri setindeki oyuncuların milliyet dağılımı verir. Verimizde milletlerin hemen hemen eşit dağıldığın gördük.



Grafik 10, Aslında burada oyuncuların mevkilerinin eşit olmayan bir şekilde dağıldığını gördük, bu sorunu veri üzerinde balance yapmak ile çözecektik ama aldığımız çıktıların istediğimiz sonuçlara yakınlığından dolayı bundan vazgeçtik. Burada yapmış olduğumuz balance işleminin başka verileri bozabileceğini düşündük.,



Grafik 11, veri setindeki Oyuncuların oynadıkları liglere göre dağılımı görüyoruz. Burada ki verilerimizin de dengeli bir dağılımda olduğunu söyleyebiliriz.

4.MODEL EĞİTİMİ

Projede piyasa değer tahmini için Random Forest ve Linear Regression modellerinin kullanılmıştır. Veri seti incelenip gerekli görülen güncellemelerin yapılmasının ardından kalan özniteliklerin vektörünün bulunduğu yeni bir sütun veri setine eklenmiştir. İlgili bölümlerin kodu Görsel 2'de gösterilmiştir. Ardından veri seti %80'e 20'lik bir oran ile eğitim ve test işlemleri için bölünmüştür. Modeller eğitim verisi ile eğitilmiş ve test verisi ile test edilmiştir. Linear Regression modeli ile 0.82'lik, Random Forest modeli ile ise 0.71'lik bir R^2 değeri elde edilmiştir.

5.SONUÇ

Proje kapsamında bir datanın setinin nasıl dağıtık hale getirileceğinin uygulamsını test etmiş olduk. Burada kullanmış olduğumuz Jupyter Notbook içinden pyspark kullanarak veriyi işlemeyi, gereksiz görülebilecek yerlerin veriden temizlenmesini, veriyi dağıtık hala getirmeyi öğrendik. Proje kapsamında amaçlarımızın başında, elimzdeki verileri kullanarak oyuncu verilerini tahmin etmeye çalıştık. Göreceli bir şekilde, manuel olarak yapılan oyuncu değeri güncellemelerininin yerine, veriyi kullanarak daha iyi şekilde tahmin edilebileceğini denedik ve gördük. Aldığımız sonuçlar bu fikrimizi belirli oranda destekler nitelikteydi.

6.KAYNAKÇA

https://stackoverflow.com/questions/36517302/randomforestclassifier-was-given-input-with-invalid-label-column-error-in-apache

https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html

https://data-flair.training/blogs/apache-spark-lazy-evaluation/

https://www.kaggle.com/code/tientd95/advanced-pyspark-for-exploratory-data-analysis

https://www.databricks.com/blog/2018/05/03/benchmarking-apache-spark-on-a-single-node-machine.html

https://medium.com/@nutanbhogendrasharma/role-of-onehotencoder-and-pipelines-in-pyspark-ml-feature-part-2-3275767e74f0

https://towardsdatascience.com/building-a-linear-regression-with-pyspark-and-mllib-d065c3ba246a

https://medium.com/@junwan01/oversampling-and-undersampling-with-pyspark-5dbc25cdf253

https://towardsdatascience.com/build-an-end-to-end-machine-learning-model-with-mllib-in-pyspark-4917bdf289c5

https://www.kaggle.com/code/eryash15/financial-fraud-detection-using-pyspark-mllib

https://www.kaggle.com/search?q=mllib+notebookLanguage%3APython

https://www.kaggle.com/code/fatmakursun/pyspark-ml-tutorial-for-beginners

https://towardsdatascience.com/big-data-visualization-using-datashader-in-python-c3fd00b9b6fc

https://towardsdatascience.com/apache-spark-mllib-tutorial-ec6f1cb336a9

https://www.kaggle.com/code/ilyapozdnyakov/house-prices-prediction-pyspark-guide

https://www.databricks.com/notebooks/gallery/GettingStartedWithSparkMLlib.html

7.GÖRSELLER

```
root
 |-- Team: integer (nullable = true)
 |-- League: string (nullable = true)
 |-- player: string (nullable = true)
 |-- Substitution: integer (nullable = true)
 |-- xG Per Avg Match: double (nullable = true)
 |-- Shots: integer (nullable = true)
 |-- OnTarget: integer (nullable = true)
 |-- Shots Per Avg Match: double (nullable = true)
 |-- On Target Per Avg Match: double (nullable = true)
 |-- value: integer (nullable = true)
 |-- nationality: string (nullable = true)
 |-- position: string (nullable = true)
 |-- squad: string (nullable = true)
 |-- goals: integer (nullable = true)
 |-- goals_per_shot: double (nullable = true)
 |-- passes_completed: integer (nullable = true)
 |-- assisted_shots: integer (nullable = true)
 |-- passes_blocked: integer (nullable = true)
 |-- touches: integer (nullable = true)
 |-- height: integer (nullable = true)
 |-- assists: integer (nullable = true)
 |-- games_starts: integer (nullable = true)
 |-- minutes: integer (nullable = true)
```

Görsel 1

```
assembler = VectorAssembler(inputCols=data_frame.drop('value').columns, outputCol="features")
data_frame = assembler.transform(data_frame)
```

```
scaler = StandardScaler(inputCol="features", outputCol="scaledFeatures", withStd=True, withMean=False)
scalerModel = scaler.fit(data_frame)
data_frame = scalerModel.transform(data_frame)
```

Görsel 2