경주마 승률 예측

컴퓨터공학과 2023208020 정그린

Contents 목차

01 개요 및 필요성

배경	경마 산업의 동향 파악 및 효율성 제고
의의 및 중요성	정확한 승률 예측은 경주마 소유자, 훈련사, 베팅자 등 다양한 이해관계자들에게 경마에 대한 투자 및 전략 수립에 도움을 줌
목표	보다 정확한 경주마 승률의 예측을 통해 경마 업계에서 사용자들에게 효율적인 의사 결정을 지원

01 개요 및 필요성

관련 내용

산업 접근성 강화

경마에 대한 접근성, 알림, 홍보

투자 최적화

경마 산업에 대한 배팅자들에 효율적인 의사 결정 과정 및 전략 수립 편의성 제공

손실 최소화

투자 위험성 낮추기 위한 대

02 데이터 설명

데이터 요소

Horse Racing

Horse Racing Data from 1990

races 열

- rid - 레이스 ID;
- 코스 - 경주 코스, 괄호 안의 국가 코드
- 시간 - hh:mm 형식의 경주 시간, London TZ;
- 날짜 - 경주 날짜.
- 제목 - 경주의 제목입니다.
- rclass - 인종 클래스;
- 밴드 - 밴드;
- 연령 - 연령 허용
- 거리 - 거리;
- 상태 - 표면 상태;
- 장애물 - 장애물, 유형 및 금액;
- 상품 - 장소 상품;
- WinTime - 표시되는 가장 좋은 시간입니다.
- 상 - 총 상금(상금 합계 열)
- 미터법 - 미터 단위의 거리;
- countryCode - 경주 국가.
- ncond - 조건 유형(조건 기능에서 생성됨)
- class - 클래스 유형(rclass 기능에서 생성됨)

horses 열

- rid - 레이스 ID;
- HorseName - 말 이름;
- 나이 - 말 나이;
- 안장 - 말이 출발하는 안장 #;
- trainingName - 트레이너 이름.
- jockeyName - 기수 이름;
- position - 마무리 위치
- positionL - 추적되는 말
- dist - 말이 승자로부터 얼마나 멀리 왔는지
- WeightSt - 말의 무게(St);
- WeightLb - 말의 무게(Lb);
- 과체중 - 과체중 코드;
- 아웃핸디캡 - 핸디캡;
- headGear - 헤드 기어 코드;
- RPR - RP 등급;
- TR - 최고 속도;
- 아버지 - 말의 아버지 이름;
- 어머니 - 말의 어머니 이름;
- gfather - 말의 할아버지 이름.
- 주자 - 총 주자;
- 마진 - 경주에 대한 소수점 가격의 합계
- 체중 - 말의 체중(kg);
- res_win - 말의 승리 여부;
- res_place - 말 배치 여부

02 데이터 설명

데이터 세트

```
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/horses_2003.csv
/kaggle/input/races_2003.csv
/kaggle/input/races_2007.csv
/kaggle/input/horses_2009.csv
/kaggle/input/horses_2020.csv
/kaggle/input/horses_2019.csv
/kaggle/input/races_2000.csv
/kaggle/input/horses_2002.csv
/kaggle/input/races_2010.csv
/kaggle/input/horses_2015.csv
/kaggle/input/horses_2017.csv
/kaggle/input/races_2012.csv
/kaggle/input/horses_2011.csv
```

```
/kaggle/input/horses_2004.csv
/kaggle/input/horses_1991.csv
/kaggle/input/horses_1997.csv
/kaggle/input/horses_2012.csv
/kaggle/input/horses_2005.csv
/kaggle/input/horses_1992.csv
/kaggle/input/horses_2014.csv
/kaggle/input/races_1992.csv
/kaggle/input/horses_2000.csv
/kaggle/input/horses_1995.csv
/kaggle/input/horses_2018.csv
```

1990~2020년까지의 경마 데이터.CSV

03 EDA 및 시각화

데이터 분석

plotPerColumnDistribution

```
def plotPerColumnDistribution(df, nGraphShown, nGraphPerRow):
    nunique = df.nunique()
    df = df[[col for col in df if nunique[col] > 1 and nunique[col] < 50]]
    nRow, nCol = df.shape
    columnNames = list(df)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num=None, figsize=(6 * nGraphPerRow, 8 * nGraphRow), dpi=80, facecolor
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation=90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad=1.0, w_pad=1.0, h_pad=1.0)
    plt.show()
```

선택한 데이터(년도)의 처음 5개 행 출력

```
df1.head(5)
```

	rid	horseName	age	saddle	decimalPrice	isFav	trainerName	jockeyName	position	positionL	...	TR	OR	father	mother	gfather	runners	margin	weight	res_win	res_place
0	271018	Combermere	6.0	0.0	0.222222	0	R G Frost	J Frost	1	NaN	...	94.0	NaN	Absalom	Queen's Parade	Sovereign Path	14	1.521003	69	1.0	1
1	271018	Royal Battery	6.0	0.0	0.090909	0	D H Barons	S Earle	2	10	...	88.0	NaN	Norfolk Air	All At Sea	Man The Rail	14	1.521003	69	0.0	1
2	271018	Just So	7.0	0.0	0.029412	0	J D Roberts	S Burrough	3	15	...	71.0	NaN	Sousa	Just Camilla	Ascertain I	14	1.521003	66	0.0	1
3	271018	Mandraki Shuffle	8.0	0.0	0.090909	0	Oliver Sherwood	M Richards	4	20	...	65.0	NaN	Mandalus	Indictment	Desert Call	14	1.521003	69	0.0	0
4	271018	Turnberry Dawn	8.0	0.0	0.047619	0	T B Hallett	P Richards	5	dist	...	45.0	NaN	Fair Turn	Shuil Alainn	Levanter	14	1.521003	69	0.0	0

Keyword

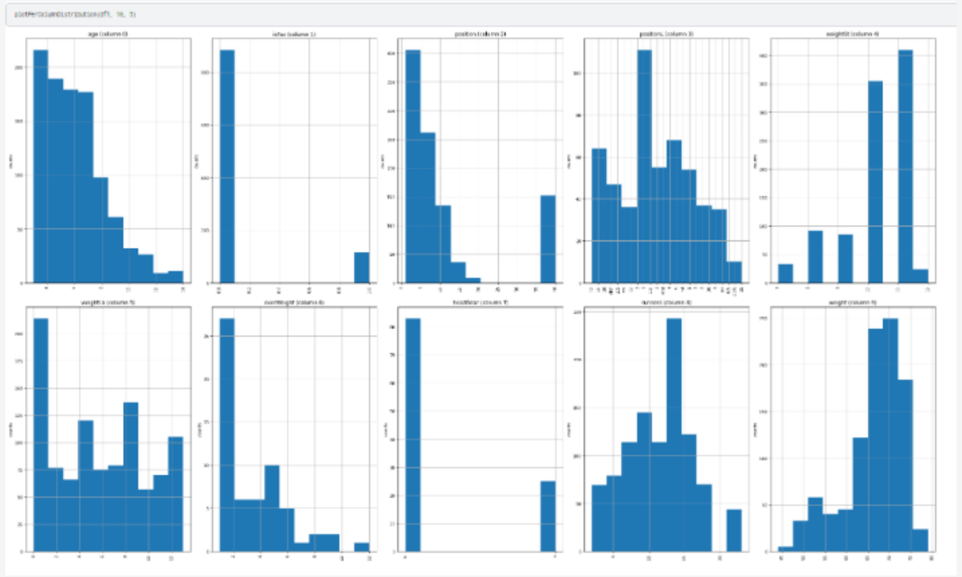
EDA (EXPLORATORY DATA ANALYSIS) 탐색적 데이터 분석

: 데이터를 분석하기 전에 그래프나 통계적인 방법으로 자료를 직관적으로 바라보는 과정

03 EDA 및 시각화

데이터 분석

plotPerColumnDistribution



나이, ISFAV, 시작 및 포착 위치, 몸무게, 헤드 기어 장착 유무 등의 열에 대한 분포 시각화 히스토그램

03 EDA 및 시각화

데이터 분석

plotCorrelationMatrix

```
def plotPerColumnDistribution(df, nGraphShown, nGraphPerRow):
    nunique = df.nunique()
    df = df[[col for col in df if nunique[col] > 1 and nunique[col] < 50]]
    nRow, nCol = df.shape
    columnNames = list(df)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num=None, figsize=(6 * nGraphPerRow, 8 * nGraphRow), dpi=80
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation=90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad=1.0, w_pad=1.0, h_pad=1.0)
    plt.show()
```

선택 년도 처음 5개 행 출력

```
df2.head(5)
```

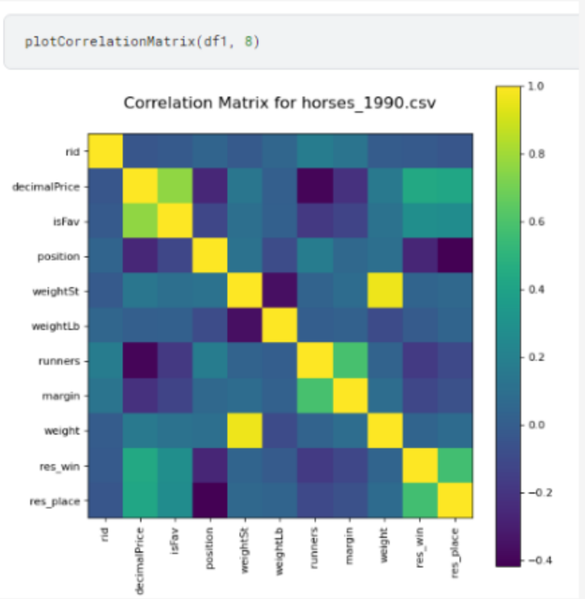
	rid	horseName	age	saddle	decimalPrice	isFav	trainerName	jockeyName	position	positionL	...	TR	OR	father	mother	gfather	runners	margin	weight	res_win	res_place
0	272927	Strong Suspicion	7.0	0.0	0.222222	1	A L T Moore	P L Malone	1	NaN	...	NaN	NaN	Strong Gale	Shady Doorknocker	Mon Capitaine	13	1.431524	69	1.0	1
1	272927	Baltray	10.0	0.0	0.066667	0	J T R Dreaper	K Morgan	2	2.5	...	NaN	NaN	Hardboy	Unsinkable Sarah	Mon Capitaine	13	1.431524	73	0.0	1
2	272927	Mister Chatterbox	10.0	0.0	0.166667	0	J R H Fowler	D P Fagan	3	10	...	NaN	NaN	Le Bavard	Farthest South	Shackleton	13	1.431524	70	0.0	1
3	272927	Culleendubh	11.0	0.0	0.111111	0	S F Maye	G Kilfeather	4	10	...	NaN	NaN	Bonne Noel	Har Valley	Harwell	13	1.431524	73	0.0	0
4	272927	Calliealla	6.0	0.0	0.066667	0	M Keane	A Powell	5	dist	...	NaN	NaN	Callernish	Shady Ahan	Mon Capitaine	13	1.431524	64	0.0	0

03 EDA 및 시각화

데이터 분석

plotPerColumnDistribution

```
plotCorrelationMatrix(df1, 8)
```



말의 몸무게, 도착 위치, 승리 여부 등의 변수들 간 관계 파악 가능

03 EDA 및 시각화

데이터 분석

plotScatterMatrix

```
def plotScatterMatrix(df, plotSize, textSize):
    df = df.select_dtypes(include=[np.number])
    df = df.dropna('columns')
    df = df[[col for col in df if df[col].nunique() > 1]]
    columnNames = list(df)
    if len(columnNames) > 10:
        columnNames = columnNames[:10]
    df = df[columnNames]
    ax = pd.plotting.scatter_matrix(df, alpha=0.75, figsize=[plotSize, pl
    corrs = df.corr().values
    for i, j in zip(*plt.np.triu_indices_from(ax, k=1)):
        ax[i, j].annotate('Corr. coef = %.3f' % corrs[i, j], (0.8, 0.2), 
    plt.suptitle('Scatter and Density Plot')
    plt.show()
```

20개 중 선택 년도 처음 5개 행 출력

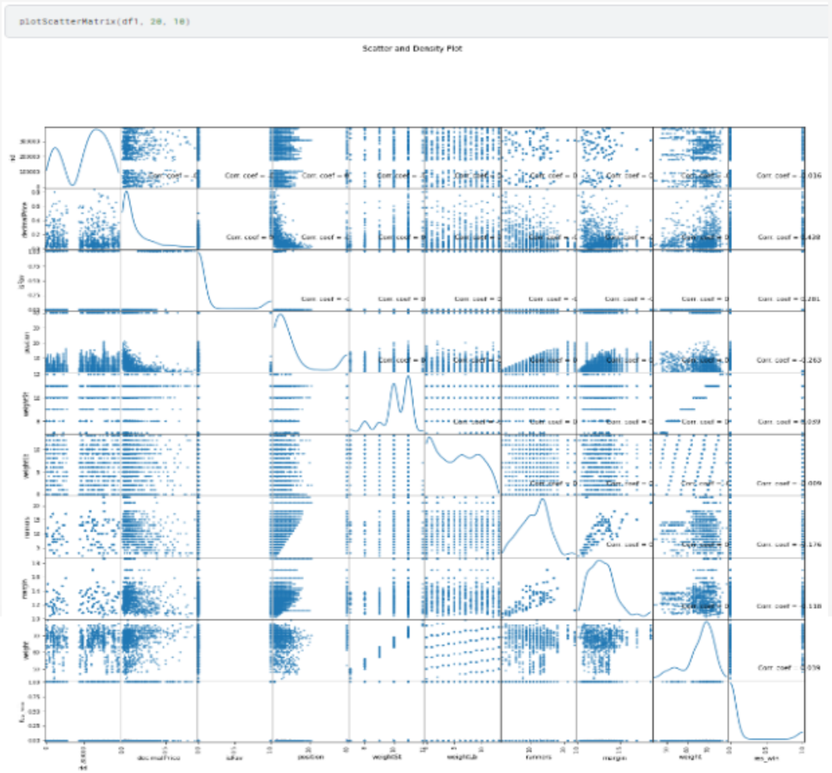
```
df3.head(5)
```

	rid	horseName	age	saddle	decimalPrice	isFav	trainerName	jockeyName	position	positionL	...	TR	OR	father	mother	gfather	runners	margin	weight	res_win	res_place
0	281414	Far Over Struy	7.0	0.0	0.692308	1	Oliver Sherwood	Jamie Osborne	1	NaN	...	104.0	NaN	Ardross	Musical Gift I	Princely Gift	3	1.070085	74	1.0	1
1	281414	Raba Riba	7.0	0.0	0.111111	0	John Spearing	Martin Lynch	2	15	...	86.0	NaN	Oats	Erica Alba	Yukon Eric	3	1.070085	72	0.0	0
2	281414	Lady Remainder	5.0	0.0	0.266667	0	P A Blockley	S Wynne	3	15	...	52.0	NaN	Remainder Man	My Aisling	John De Coombe	3	1.070085	61	0.0	0
3	282301	Brig's Gazelle	10.0	0.0	0.166667	0	I Park	N Old Smith	1	NaN	...	80.0	95.0	Lord Nelson	Cliburn New Cut	New Brig	8	1.153773	61	1.0	1
4	282301	Blakes Son	7.0	0.0	0.090909	0	Michael Easterby	Lorcan Wyer	2	8	...	94.0	110.0	Blakeney	Susanna	Nijinsky	8	1.153773	70	0.0	1

03 EDA 및 시각화

데이터 분석

plotPerColumnDistribution



수치화된 열들 간 상관 관계

04 데이터 전처리

데이터 가공

```
nRowsRead = 1000
df1 = pd.read_csv('/kaggle/input/horses_1990.csv', delimiter=',', nrows=n
df1.dataframeName = 'horses_1990.csv'
nRow, nCol = df1.shape
print(f'There are {nRow} rows and {nCol} columns')
```

(기존 코드에서)

```
df = df.drop(['course_type', 'weather'], axis=1)


dfs.append(df)
```

1. 코스 조건, 날씨 요소 제거

```
dfs = []
for year in range(2000, 2021):
    file_path = os.path.join(data_path, f'horses_{year}.csv')
    df = pd.read_csv(file_path, delimiter=',')
    df.dataframeName = f'horses_{year}.csv'
    dfs.append(df)
```

```
for year, df in zip(range(2000, 2021), dfs):
    nRow, nCol = df.shape
    print(f'Year {year}: There are {nRow} rows and {nCol} columns')
```

2. 2000년대부터의 데이터만을 적용하여 정확도를 향상

05 모델 학습 및 평가

훈련 및 성능 평가

```
from sklearn.linear_model import LinearRegression
```

선형 회귀 모델 생성

+

변수 간 상관 관계 파악

+

평가 지표 제시

```
def predict_win_rate(df):

    df['win_rate'] = df['won'] / df['total_races']

    selected_columns = ['horse_age', 'horse_rating', 'jockey_wins', 'jockey_races', 'trainer_wins', 'trainer_races', 'win_rate']
    df = df[selected_columns].dropna()

    X = df.drop('win_rate', axis=1)
    y = df['win_rate']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    model = LinearRegression()
    model.fit(X_train_scaled, y_train)

    y_pred = model.predict(X_test_scaled)

    mse = mean_squared_error(y_test, y_pred)
    print(f'Mean Squared Error: {mse}')
```

```
Mean Squared Error (MSE): 0.92
```

```
dfs = []
for year in range(2000, 2021):
    file_path = os.path.join(data_path, f'horses_{year}.csv')
    df = pd.read_csv(file_path, delimiter=',', nrows=1000)

for year, df in zip(range(2000, 2021), dfs):
    print(f'\nYear {year}')
    predict_win_rate(df)
```

05 모델 학습 및 평가

비교 모델

```
temp_cols=df.columns.tolist()
index=df.columns.get_loc("finish_time")
new_cols=temp_cols[index:index+1] + temp_cols[0:index] + temp_cols[index+1:]
df=df[new_cols]
```

시간 예측 모델

- 연속적인 수치 도출
- 등수 값 표시

```
df_0['pred']=voting_rg.predict(X_0)

df_0 = df_0[['finish_time','pred','result']]

df_0['result_pred'] = df_0['pred'].rank(ascending=True).astype(int)

df_0
```

	finish_time	pred	result	result_pred
0	83.92	84.100102	10	3
1	83.56	84.568903	8	10
2	83.40	84.163158	7	4
3	83.62	84.429591	9	9
4	83.24	84.419722	6	8
5	82.83	84.417288	3	7
6	84.15	84.781629	12	12
7	82.64	84.076442	1	2
8	84.20	84.332607	13	6
9	92.20	85.237954	14	14
10	82.77	83.656561	2	1
11	82.98	84.636649	4	11
12	83.94	84.253453	11	5
13	83.08	84.860659	5	13

```
df_3['pred']=voting_rg.predict(X_3)

df_3 = df_3[['finish_time','pred','result']]

df_3['result_pred'] = df_3['pred'].rank(ascending=True).astype(int)

df_3
```

	finish_time	pred	result	result_pred
42	69.23	84.674148	5	8
43	69.34	84.531937	6	7
44	69.16	84.068350	2	2
45	70.22	84.405887	11	6
46	68.89	83.110261	1	1
47	69.75	84.769281	10	11
48	69.22	84.733355	4	10
49	69.57	84.135625	7	3
50	69.67	84.885198	9	12
51	69.20	84.161885	3	4
52	69.66	84.323752	8	5
53	70.52	84.697715	12	9

06 결론 및 소감

01

타 스포츠 종목과 비교해 참고자료 부족

02

평소 생소한 경마에 대한 지식 부족

03

변수들 간 상관관계 파악 이해 어려움

04

방대한 데이터 가공하며 데이터 분석에 흥미

Thank you

Any Question?