

Project: The Android App Market on Google Play

Task 3: Instructions

- Convert `Installs` and `Price` columns to `float` data type using `astype()` function.
- Verify the corrected data types by using the `dtypes` attribute of `apps` dataframe. (You can also reuse the `info()` function to verify the corrected data types)

Helpful links:

- `astype()` [documentation](#)
- [Pandas data types](#)

Take Hint

Previous Task Next Task

3. Correcting data types

From the previous task we noticed that `Installs` and `Price` were categorized as object data type (and not `int` or `float`) as we would like. This is because these two columns originally had mixed input types: digits and special characters. To know more about Pandas data types, read [this](#).

The four features that we will be working with most frequently henceforth are `Installs`, `Size`, `Rating` and `Price`. While `Size` and `Rating` are both `float` (i.e. purely numerical data types), we still need to work on `Installs` and `Price` to make them numeric.

```
In [0]: import numpy as np

# Convert Installs to float data type
apps[...] = apps[...].astype(...)

# Convert Price to float data type
apps[...] = apps[...].astype(...)

# Checking dtypes of the apps dataframe
print(...)
```

4. Exploring app categories

Check Project

Google Play Store Analysis - Task 3

Task 3 Instructions

- Convert `Installs` and `Price` columns to `float` data type using the `astype()` function.
- Verify the corrected data types by using the `dtypes` attribute of the `apps` DataFrame. You can also reuse the `info()` function to verify the corrected data types.

Correct Code Implementation

```
# Step 1: Convert 'Installs' column to float
apps['Installs'] = apps['Installs'].astype(float)
```

```
# Step 2: Convert 'Price' column to float
apps['Price'] = apps['Price'].astype(float)
```

```
# Step 3: Verify the corrected data types
print(apps.dtypes)
```

Explanation of the Code

- **Convert to float**:**
 - The `.astype(float)` method converts the data type of the specified

column to `float`.

- This step ensures numerical operations can be performed on these columns in subsequent tasks.

2. **Verify data types**:

- The `.dtypes` attribute displays the data types of all columns in the DataFrame.

- Optionally, you can also use `apps.info()` to inspect the data types.